VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Vzorkování lokálních struktur rozsáhlých síťových dat

Sampling Local Structures of Large Network Data

2018 Bc. Jakub Plesník

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Jakub Plesník**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Vzorkování lokálních struktur rozsáhlých síťových dat
Sampling Local Structures of Large Network Data

Jazyk vypracování: čeština

Zásady pro vypracování:

Analýza tzv. velkých dat je současným fenoménem, kterým se zabývají nejenom vědci, ale také firmy, které mohou díky analýze firemních dat či např. analýzou dat reprezentující chování zákazníků plánovat své budoucí strategie. Pokud je dat velké množství, můžeme místo celé datové kolekce pracovat jen s jejím reprezentativním vzorkem. Cílem práce je implementace algoritmů pro tzv. vzorkování (sampling) dat představujících reálné datové kolekce (jako je Web, Internet, sociální sítě). Vzorkovací metody budou metodami tzv. back-in-time vzorkování, jehož cílem je napodobit stav sítě (grafu) v určitém časovém okamžiku jejího vývoje vhodným vzorkem.
1. Seznamte se s problematikou komplexních sítí.
2. Seznamte se základními přístupy k vzorkování (samplingu) datových kolekcí reprezentujících reálné sítě.
3. Seznamte se s nejčastěji používanými algoritmy samplingu a proveďte jejich srovnání.
4. Vyberte vhodné algoritmy pro back-in time vzorkování a naimplementujte je.
5. Navrhněte experimenty a nad zvolenými datovými kolekcemi je proveďte. Experimenty vyhodnoťte.

Seznam doporučené odborné literatury:

[1] M. E. J. Newman, Networks: An Introduction, Oxford University Press (2010), ISBN-10: 0199206651.
[2] Pili Hu, Wing Cheong Lau, A Survey and Taxonomy of Graph Sampling http://arxiv.org/abs/1308.5865
[3] J. Leskovec, Ch. Faloutsos, 2006. Sampling from Large Graphs In proc. of the KDD '06, pp. 631636
[4] Podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **RNDr. Eliška Ochodková, Ph.D.**

Datum zadání: 01.09.2017
Datum odevzdání: 30.04.2018

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 26. dubna 2018 .....

Chtěl bych poděkovat paní RNDr. Elišce Ochodkové, Ph.D. za odborné vedení, trpělivost a ochotu, kterou mi v průběhu zpracování diplomové práce věnovala.

Abstrakt

Diplomová práce popisuje teorii grafů, vlastnosti sítí, vzorkovací algoritmy a metodu vzorkování back-in-time. Práce se věnuje temporálním sítím, uvádí příklady reálných sítí a popisuje reálné datové sady, které byly použity v rámci experimentů. Hlavním cílem této diplomové práce jsou experimenty spočívající v hledání nejlepší vzorkovací metody. Experimenty jsou prováděny nad sítěmi generovanými pomocí modelů i nad reálnými sítěmi. Nejlepší vzorkovací metoda je vyhodnocována porovnáváním kvality generovaných vzorků. Vzorkovací metody byly vybrány tak, aby pokryly všechny základní přístupy, ale největší zastoupení připadá na algoritmy založeny na náhodné procházce.

Klíčová slova: grafy, reálné sítě, vzorkování v čase, temporální sítě,

Abstract

This Master's thesis describes graph theory, attributes of network, sampling alghoritms and a back-in-time sampling method. This thesis is dedicated to the temporal networks and provides examples of real networks and describes real datasets that were used in experiments. The main goals of this thesis are experiments providing information about the best sampling alghoritm. For the purpose of experiments, generated models and real datasets are used. The best sampling method is evaluated by comparison of sample quality. Sampling alghoritms were selected to cover all basic principles, but most of them belong to a group based on random walk.

Key Words: graph, networks, sampling, back in time, temporal networks

Obsah

Seznam použitých zkratek a symbolů

API	–	Application Programming Interface
AS	–	Autonomous systems
BFS	–	Breadth-first search
DG	–	Top Degree
FF	–	Forest Fire
MHRW	–	Metropolis Haistings Random Walk
PPIN	–	Protein-Protein Interaction Network
RE	–	Random Edge
RN	–	Random Node
RW	–	Random Walk
UI	–	User Interface
WPF	–	Windows Presentation Foundation

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Sítě nám pomáhají napříč různými obory a mohou modelovat osoby nebo objekty z reálného světa a vazby mezi nimi. Tato reprezentace dat a její analýza nám může přinášet spoustu informací, které jsou běžnou reprezentací dat skryty. Výsledná data je možné promítnout do zpeněžitelné formy. Právě proto je v současné době analýza dat tak velký fenomén. Analýza síťových dat umožňuje vědcům a firmám např. predikovat trh, politickou předvolební scénu a modifikovat strategie na základě výsledků. Rozsah dat neumožňuje vždy pracovat s celými datovými sadami, a proto je využíváno takzvaných vzorků. Pro získání vzorků využíváme tzv. vzorkovací algoritmy.

Cílem této práce je implementace aplikace pro řadu experimentů, spočívajících ve vzorkování pomocí různých vzorkovacích algoritmů metodou back-in-time a vyhodnocení kvality výsledných vzorků nad různými datovými soubory.

Tato práce popisuje základní pojmy z oblasti síťových dat, teorie grafů a jednotlivé vlastnosti, jenž u sítí sledujeme. V práci jsou popsány vzorkovací algoritmy, které jsou implementovány v rámci vytvořené aplikace. Práce obsahuje popis aplikace, která byla využita pro provedení experimentů nad různými datovými sadami s metodou vzorkování back-in-time. Následně se práce věnuje prezentaci výsledků experimentů. V závěru je porovnán výkon jednotlivých vzorkovacích algoritmů.

2 Reálné sítě

Reálné sítě vznikají v různých odvětvích lidské činnosti od informačních technologií přes geografii, sociologii až po biologii. Příklady takových sítí mohou být sítě přátelství na sociálních sítích (Facebook, Instagram, Tinder), kde mapujeme uživatele dané sociální sítě jako entity a vazby mezi nimi vytváříme na základě existence přátelství, sledování nebo vzájemné sympatie podle kontextu dané sociální sítě. Taková síť může sloužit např. k vytváření strategických modelů pro marketingové účely firem nebo průzkumu zájmů či názorů.

Sítě komunikace v rámci pracovního prostředí, nebo komunikace v prostředí některého z instantních komunikátorů, znázorňují osoby jako entity sítě a vazba mezi dvěmi osobami je v případě, že mezi nimi proběhla komunikace. Taková síť může popisovat vnitrofiremní stuktury nebo chování uživatelů komunikačních služeb.

V oblasti biologie můžeme pomocí sítí např. reprezentovat vzájemnou interakci proteinů (PPIN). PPIN znázorňuje základní proces probíhající v buňkách a usnadňuje pochopení chování buňky za normálních podmínek nebo při napadení nemocí. Informace získané z PPIN umožňují např. přiřadit vlastnosti dosud necharakterizovaným proteinům nebo charakterizovat vztahy mezi proteiny, které společně tvoří multi-molekulární komplex jako jsou proteazomy [34].

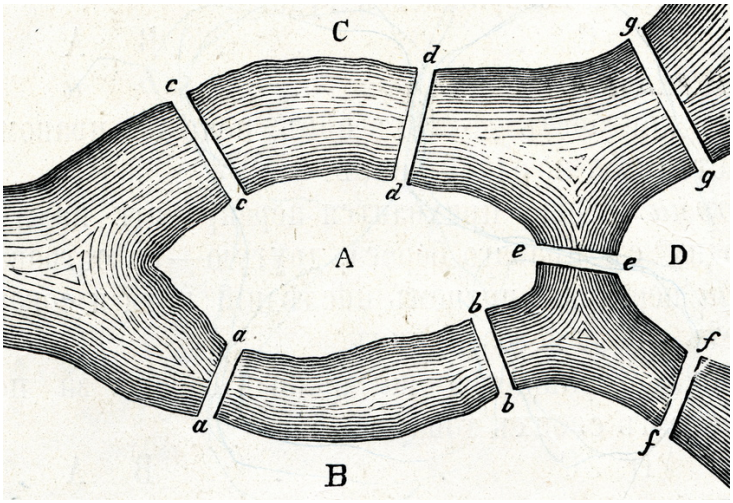
Topologické sítě zastupuje například síť internetu, kde jsou jako vrcholy počítače, servery nebo routery. Hrana je mezi entitami vytvořena v případě, že mezi nimi proběhla komunikace pomocí posílání paketů. Znalosti vyplývající z analýzy takové sítě mohou např. odhalit informace o zranitelnosti sítě proti náhodnému výpadku nebo cílenému útoku.

U reálných sítí můžeme sledovat velké množství vlastností, které společně popisují strukturu sítě. Pokud se jedná o rozsáhlou síť, jejíž struktura je nepravidelná, komplexní a vyvíjí se ve vztahu k času, mluvíme o takzvané **komplexní síti**. U těchto sítí není snadné odvodit chování sítě jako celku pouze ze znalostí chování jednotlivých entit.

K analýze síťových dat vědci využívají znalostí z oboru nazývaného **teorie grafů**. Jedná se o obor diskrétní matematiky, který zkoumá vlastnosti grafů jakožto matematickou reprezentaci sítě.

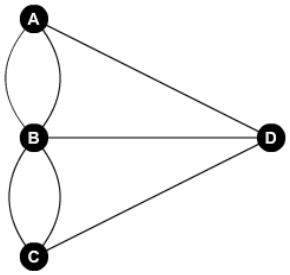
2.1 Teorie grafů

Tento obor datuje svůj počátek do roku 1735, v Königsbergu hlavním městě východního Pruska, jenž byl v tehdejší době vzkvétajícím městem a centrem obchodu, postavili 7 mostů mezi dvěma břehy řeky Pregel a ostrovem, který byl řekou obklopen viz obr. 1 . Rozložení mostů vytvořilo zajímavou hádanku, která zní: Je možné vytvořit cestu, která povede přes všechny mosty tak, aby žádný nebyl navštíven dvakrát? Tento problém nenacházel řešení, dokud švýcarský matematik Leonard Euler nepřišel s matematickým důkazem, že taková cesta neexistuje.



Obrázek 1: Skica 7 mostů nad řekou Pregel [32]

Euler vytvořil první graf, kde každá část pevniny byla reprezentována písmeny a mezi těmito potencionálními vrcholy byly vytvořeny cesty podle mostů, jenž tyto část propojovaly. Tento graf je uveden na obrázku 2. Na základě této reprezentace došel k závěru, že vrchol, jenž má lichý počet připojení musí být počátečním nebo konečným bodem v cestě, protože při návštěvě takového vrcholu nemusí být žádná cesta, která zůstane pro jeho opuštění.



Obrázek 2: Reprezentace problému 7 mostů pomocí grafu

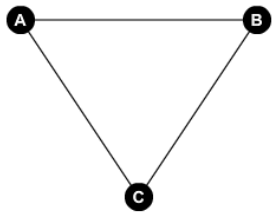
Eulerův graf znázorňující problém 7 mostů obsahuje 4 vrcholy s lichým stupněm. Proto nemůže existovat cesta, která by splňovala podmínku navštívení každé cesty pouze jednou. Eulerův důkaz byl prvním případem použití grafu pro nalezení řešení matematického problému [1].

2.2 Základní pojmy

Jednotlivé entity, které se nachází v sítích nazýváme **vrcholy**. Propojení neboli vazby mezi nimi nazýváme **hranou**. Tato definice nám umožňuje dívat se na sítě jako na grafy. Dle definice je **graf**(také jednoduchý graf nebo prostý graf) uspořádaná dvojice $G = (V, E)$, kde V je neprázdná množina vrcholů a E je množina hran. Hrana je tvořena dvouprvkovou podmnožinou množiny

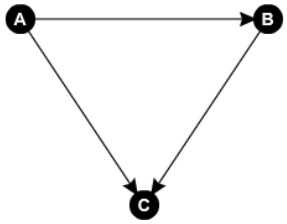
vrcholů V. Hranu tedy můžeme zapsat jako dvojici (u, v). Z definice prostého grafu vyplývá, že mezi dvěmi libovolnými vrcholy může existovat pouze jedna hrana [2].

Neorientovaný graf vychází ze základní definice s tím, že hrana je neuspořádaná dvouprvková podmnožina $\{u, v\}$ z množiny vrcholů V. Maximální počet hran v neorientovaném jednoduchém grafu je $n(n-1)$ [2]. Příklad grafu je na obr. 3.



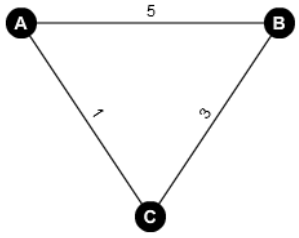
Obrázek 3: Neorientovaný graf

Orientovaný graf také vychází ze základní definice s tím, že hrana je uspořádaná dvojice (u,v) prvků $u, v \in V$, kde V je množina vrcholů. Uspořádáním dvojice prvků dosáhneme orientace hrany. To znamená, že (u,v) není stejná jako (v,u), ale jedná se o hrany mezi stejnými vrcholy s odlišnou orientací. Příklad grafu je na obr. 4, orientace hrany je znázorněna směrem šipky [2].



Obrázek 4: Orientovaný graf

Ohodnocený graf rozšiřuje definice orientovaného nebo neorientovaného grafu, kde každé hraně (u, v) přiřazuje reálné číslo w, které je označováno jako váha [1, 2]. Příklad ohodnoceného grafu je na obr. 5. Váha příslušné hrany je vyjádřena číslem nad hranou.



Obrázek 5: Vážený graf

2.3 Vlastnosti sítí

V této části budou popsány základní vlastnosti grafů a uvedeny jejich definice.

2.3.1 Stupeň, průměrný stupeň a distribuce stupňů

Stupeň je jedním z nejzákladnějších atributů každého vrcholu a vyjadřuje počet hran, se kterými je daný vrchol incidentní.

Na příkladu sociálních sítích stupeň pro konkrétní vrchol (entitu) znamená počet přátel (Facebook) nebo počet spojení (LinkedIn). Mezi stupni a počtem hran v grafu je přímý vztah. Počet hran v grafu $|E|$ může být vyjádřen jako součet stupňů všech vrcholů v grafu děleno dvěma, jelikož každá hrana je započtena dvakrát. Tento vztah je vyjádřen v rovnici 1 kde $|E|$ je počet hran, N je množina vrcholů a k označuje stupeň vrcholu i.

$$|E| = \frac{1}{2}\sum_{i=1}^{N} k_i \tag{1}$$

V případě orientovaných sítí je situace složitější a kromě obecného stupně započítávajícího všechny hrany můžeme počítat tzv. vstupní stupeň, což je součet hran směřujících do vrcholu a výstupní stupeň, součet hran z vrcholu vystupujících. V anglické literatuře se setkáme s terminologií in/out degree.

V případě sociální sítě typu Twitter nebo Instagram je velikost stupně měřitelná počtem uživatelem sledovaných účtů(k_{out}) a počtem daného uživatele sledujících účtů(k_{in})

Důležitým parametrem sítě je **průměrný stupeň** $\langle k \rangle$, který lze vyjádřit jako sumu stupňů všech vrcholů děleno počtem vrcholů. Vyjádřeno rovnicí 2, kde N je počet vrcholů a k_i je stupeň vrcholu i [1].

$$\langle k \rangle = \frac{k_1 + k_2 + ... + k_N}{N} = \frac{1}{N}\sum_{i=1}^{N} k_i \tag{2}$$

Distribuce stupňů p_k poskytuje pravděpodobnost, že náhodně vybraný vrchol grafu má stupeň k. Distribuce stupňů může být zkonstruována tak, že pro každý stupeň k v grafu získáme

počet vrcholů s tímto stupněm. Následně jsme schopni vytvořit graf, kde na ose Y bude počet vrcholů a na ose X stupně vrcholu. Pro získání pravděpodobnosti p_k je nutné data normalizovat [1]. Dalším praktickým zobrazením distribuce stupňů je kumulativní četnost stupňů, která umožňuje rozhodnout kolik počtů pozorování leží pod (nebo nad) určitou hodnotou v datové sadě.

2.3.2 Cesty

Vzdálenost a cesta mezi vrcholy je zásadním aspektem v technicky orientovaných sítích. V rámci teorie grafů však můžeme řešit i zdánlivě obtížné otázky jako je například vzdálenost dvou webových stránek nebo vzdálenost dvou uživatelů sociální sítě.

Graf na n vrcholech, které jsou spojeny po řadě $n-1$ hranami, se nazývá cesta a značí se P_n [2]. Rovnice 3 vyjadřuje cestu P_n mezi vrcholy v_0 a v_n. P_n může být vyjádřena jako seznam hran, kde v_x označuje vrchol grafu [1].

$$P_n = (v_0, v_1), (v_1, v_2), (v_2, v_3), ..., (v_{n-1}, v_1) \tag{3}$$

Délka cesty mezi dvěmi vrcholy v rámci teorie grafů nereflektuje fyzickou vzdálenost, ale počet hran, které cesta obsahuje. Například vzdálenost mezi dvěmi lidmi v rámci sociální sítě odkazuje na počet vazeb, přes které jsou dva určité lidé propojeni.

Nejkratší cesta mezi vrcholy u a v je taková, která obsahuje nejmenší počet hran. V grafu můžeme najít více nejkratších cest mezi dvěmi vrcholy, které budou stejné délky. Hledání nejkratší cesty může být velice obtížné především v rozsáhlých sítích, v takovém případě využíváme algoritmu procházení grafu do šířky) [1]. Pod termínem **vzdálenost** vrcholů u a v je myšlena nejkratší cesta mezi těmito vrcholy [2].

Průměrná vzdálenost neboli průměrná délka cesty je průměrná vzdálenost cesty mezi všemi páry vrcholů v síti. Tato vlastnost souvisí s takzvaným **Small World** efektem, který říká, že i přes relativně velký počet vrcholů v síti je průměrná vzdálenost l relativně malá, případně že s rostoucím počtem vrcholů roste průměrná vzdálenost velice pomalu jak popisuje vzorec 4.

$$l \propto lnN \tag{4}$$

V případě sítí splňující definici bezškálových sítí 2.4.2 je tento růst ještě pomalejší 5.

$$l \propto lnlnN \tag{5}$$

Diameter neboli **průměr sítě** je nejdelší z nejkratších cest mezi všemi vrcholy v síti.

2.3.3 Centra

Vrcholy, které mají velký význam ve struktuře sítě označujeme za centra. Pokud mluvíme o vlastnosti nazýváme ji centralita. Existuje mnoho způsobů jak centralitu matematicky vyjádřit. Nejjednodušším způsobem je určení centrality na základě stupně vrcholu (Degree centralita). Dle mnoha pozorování se v reálných sítích vyskytuje malé, přesto významné, množství vrcholů s nadprůměrně vysokým stupněm vrcholů. Centra někdy také označujeme jako huby, protože se starají o propojování různých částí struktury sítě.

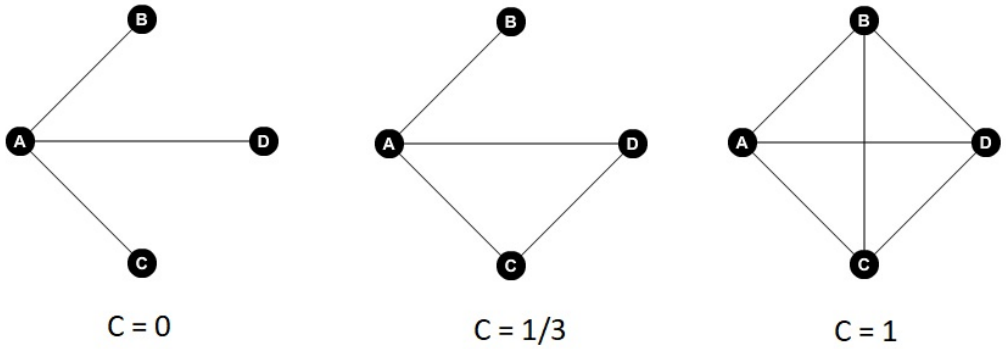
Centra nalezneme na sociálních sítích, kde to mohou být například veřejně známé osoby s vysokým počtem sledujících. V rámci sítě internetu budou centra tvořeny významnými poskytovateli internetu [1].

2.3.4 Komunity a shlukovací koeficient

Komunita může být obecně chápána jako skupina osob, které mají silnější vazby s dalšími členy komunity. Příklad komunity, který uvádí ve své knize Barabassi Albert, může být bikulturní společnost v Belgii, kde 59% obyvatel jsou Vlámové a 31% Valoni. Vlámové mluví dialektem Nizozemštiny, naproti tomu Valoni mluví Francouzsky. I přes tuto rozdílnost je Belgie stabilní zemí od roku 1830. Analýza ukázala, že Belgičané telefonicky mnohem častěji komunikují s členy téže komunity v níž se nacházejí než mimo ni [1].

Komunity jsou lokálně hustě propojené podgrafy v síti. Vrcholy uvnitř komunity musí být dosažitelné přes jiné vrcholy uvnitř komunity a zároveň mají vyšší pravděpodobnost existence hrany s vrcholy uvnitř komunity než mimo ni [1].

Shlukovací koeficient je vlastnost vrcholu, která zachycuje stupně vrcholů, ke kterým je měřený vrchol připojen. Jinými slovy shlukovací koeficient měří hustotu lokálního propojení sítě v okolí vrcholu [1].



Obrázek 6: Příklad výpočtu clustering coeficientu pro vrchol A

Lokální hodnota shlukovacího koeficientu C_i pro vrchol i lze spočítat podle rovnice 6, kde k_i je stupeň vrcholu i a L_i je počet hran mezi sousedy vrcholu i [1].

$$C_i = \frac{2L_i}{k_i(k_i - 1)} \tag{6}$$

Globální hodnota, často uváděna jako průměrný shlukovací koeficient, je vypočítávána podle rovnice 7 [1].

$$\langle C \rangle = \frac{1}{N} \sum_{i=1}^{N} C_i \tag{7}$$

2.4 Modely sítí

Pro napodobení některých vlastností reálných sítí vznikají tzv. **modely**. Model můžeme chápat jako algoritmus, který je schopen vygeneroval síť s určitými charakteristickými vlastnostmi, jenž definuje daný model. V rámci experimentů jsou i tyto modely podrobeny analýze.

2.4.1 Erdős–Rényi model

Jedná se o model, který navrhli matematici Paul Erdős a Alfréd Rényi. Zapisujeme jako $G(n, m)$. Tento model je založen na pevně definovaném počtu vrcholů n, které jsou následně s uniformní pravděpodobností propojovány pomocí hran, počet hran je pevně definován jako parametr m. Erdős–Rényi model byl zveřejněn v roce 1959, současně a nezávisle však vznikl model, jehož autorem je matematik Edgar Gilbert. Jeho model $G(n, p)$ má také pevně definovaný počet počet vrcholů n a pravděpodobnost existence hrany mezi dvěma vrcholy p. Ekvivalentně všechny grafy s n vrcholy a m hranami mají pravděpodobnost vyjádřenou vztahem 8. Počet hran v tomto modelu se nachází průměrně $\binom{n}{2}p$, nikoliv přesně m jako v případě Erdős–Rényi [6, 7].

$$p^m(1-p)^{\binom{n}{2}-m} \tag{8}$$

2.4.2 Barabási-Albert model

Tento model je pojmenován po autorech, jimiž jsou Maďarští fyzici Albert-László Barabási a Réka Albert. Pro model je vstupem souvislý graf G_0, tedy takový, ve kterém pro každé dva vrcholy u a v z množiny vrcholů V existuje cesta. Počet vrcholů v grafu G_0 je roven m_0. Dalším vstupem jsou parametry m a n. Parametr m označuje počet hran připojovaných pro nový vrchol a musí splňovat $m \leq m_0$. Parametr n označuje požadovanou velikost grafu.

Postup: Vygenerujeme nový vrchol u, který připojíme k m jiným vrcholům v síti s pravděpodobností úměrnou jejich stupni. Tímto způsobem připojujeme nové vrcholy dokud není počet vrcholů v sítí roven n [26]. Vrcholy jsou připojovány pomocí tzv. **preferenčního připojování**. Tato vlastnost nám říká, že vrcholy, které přidáváme do grafu, budou s vyšší pravděpodobností

připojeny k vrcholům s vyšším stupněm. V rovnici 9 je $\Pi(k_i)$ pravděpodobnost připojení nové hrany k vrcholu i. Tato pravděpodobnost je závislá na stupni vrcholu i vyjádřeno jako k_i. Ve jmenovateli je součet stupňů všech vrcholů v grafu [1].

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j} \tag{9}$$

Síť, kterou označujeme za **bezškálovou** je taková,s jejíž distribuce stupňů splňuje mocninný zákon 10 s mocninným exponentem γ typicky v intervalu $\langle 2, 3\rangle$, ale není pravidlem. Bezškálové sítě při změně velikosti zachovávají některé vlastnosti jako například průměr sítě [1].

$$P(k) \sim k^{-\gamma} \tag{10}$$

Model generuje bezškálovou síť využívající preferenčního připojování. Model napodobuje vlastnosti reálných sítí jako jsou existence shluků, existence center a mocninné rozdělení stupňů. BA model splňuje definici bezškálové sítě s mocninným exponentem $\gamma = 3$ [1]. Výsledný graf je souvislý a vrcholy s vyšším stupněm mají často vysokou centralitu (closeness, betweness) [26].

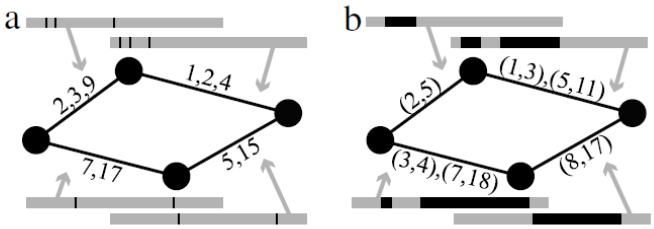
$$P(k) \sim k^{-3} \tag{11}$$

V sítích generovaných BA modelem, ale i dalších bezškálovách sítích, nalezneme vrcholy s vysokým stupněm. Ty nám tvoří takzvaná centra a propojují se s vrcholy stupně nižšího. Díky této vlastnosti je síť velice odolná proti náhodnému odebírání vrcholů, jelikož vrcholů s vysokým stupněm je málo, pravděpodobnost jejich odstranění je nízká. V případě, že by takový vrchol byl náhodně vybrán máme stále vysokou pravděpodobnost, že graf zůstane souvislý [8].

2.5 Temporální sítě

Sítě, které nacházíme v reálném světě a které potřebujeme zkoumat a analyzovat, jsou různého charakteru a pocházejí z různých oblastí. Tyto sítě se mohou v čase vyvíjet a mohou se měnit jejich vlastnosti. Pro zachycení takového vývoje využíváme temporální sítě. Temporální síť je specifický typ sítě, který se vyznačuje přidělením časové složky k hraně. Hrana či vrchol neexistuje po celou dobu, ale až od určitého časového okamžiku případně v určitém časovém intervalu.

Pro potřeby temporálních sítí musíme rozšířit základní definici stanovenou v 2.2. Hrany jsou v temporálním grafu tvořeny dvouprvkovou podmnožinou z množiny vrcholů a doplněny o časovou složku, což můžeme zapsat jako $\{x, y, t_0, t_d\}$, kde t_0 je čas výskytu hrany a t_d je trvání existence této hrany [9]. Tento formát odpovídá obrázku 7 (b), kde šedý pruh vyznačuje čas a černé bloky jsou aktivní okamžiky dané hrany [21].

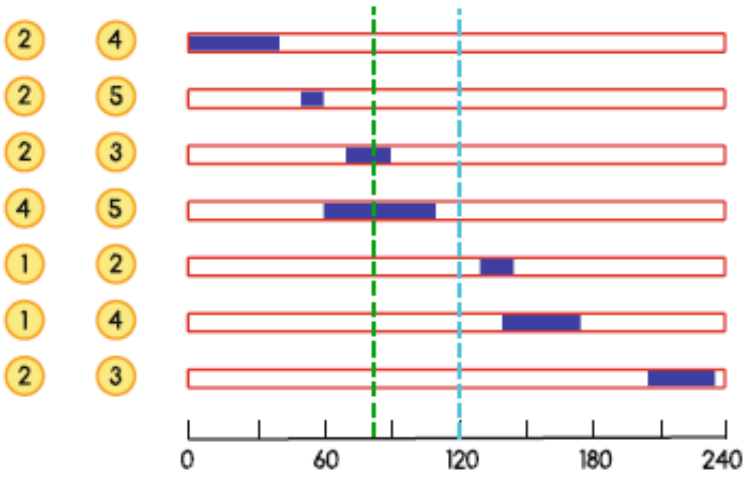


Obrázek 7: Reprezentace typů temporálních sítí [21]

V případě některých sítí, kde trvání vazby není časově vymezeno na určitý interval, je možné zjednodušit hranu jako $\{x, y, t_0\}$. To nastává především v případech, kdy trvání činnosti jenž hrana reprezentuje, nehraje zásadní roli, jako například v síti zachycující sdílení příspěvků na sociální síti [9]. Časovou složku t_0 nazýváme časovým razítkem nebo časovým okamžikem.

2.5.1 Příklady

Příkladem temporální sítě může být síť znázorňující komunikaci mezi uživateli pomocí osobní(myšleno rozhovor tváří v tvář), emailové nebo telefonické komunikace. Takový dataset by obsahoval vrcholy označující uživatele a hranu v případě, že mezi uživateli byla zaslána zpráva, uskutečněn hovor atd, spolu s časem v jakém byla akce uskutečněna. V případě telefonické nebo osobní komunikace můžeme uvažovat i nad délkou trvání takové komunikace.



Obrázek 8: Reprezentace trvání hran mezi vrcholy [9]

Na obrázku 8 je znázorněn kontakt mezi 5 osobami v rámci 4 hodinového pozorování. Plné výseky vyznačují počátek a trvání kontaktu. Přerušované čáry jsou konkrétní snapshoty neboli obrazy v konkrétním čase.

Typickým příkladem topologické temporální sítě je síť internetu, kde putující pakety vytváří komunikační síť, která se velmi rychle mění. Zástupcem této skupiny sítí je například datový soubor Autonomních systémů 5.3.6.

Ekologické sítě zachycují interakce mezi různými živočišnými druhy nebo organismy. Příkladem může být síť znázorňující tropické druhy zvířat a jejich potravní řetězec ve smyslu kdo je čí kořist [21]. Taková síť by se zcela jistě dala znázornit staticky, ale v některých případech by mohlo dojít ke ztrátě informace. V Africe, kde se střídají období monzunů a období sucha, mohou některá zvířata jako například lvi preferovat rozdílnou potravu [18] v závislosti na ročním období. Tato informace by se dala modelovat s využitím temporální sítě, pokud bychom jako časovou složku použili roční období nebo detailnější údaj např. měsíce v roce, pokud to data umožní.

Dalšími příklady mohou být sítě přátelství, mapující datum vzniku vazby mezi dvěmi aktéry na sociální síti, sítě spolupráce v rámci oboru nebo sítě proteinové interakce [21].

3 Vzorkování

Vzorkování je obecně činnost, kde se snažíme rozsáhlý soubor nahradit menším nebo snadněji zpracovatelným. Například při volebních průzkumech, se pouhá skupina lidí čítající řádově tisíce jedinců může z určitého hlediska považovat za reprezentativní vzorek. V oboru elektrotechniky se můžeme také setkat s pojmem vzorkování, například vzorkování signálu je metoda vytvoření množiny diskrétních bodů, jež budou reprezentovat určitý úsek spojitého signálu. Analogicky je využíváno i vzorkování nad síťovými daty, kde se z důvodů níže popsaných snažíme vytvořit podgraf mající určité vlastnosti. Takový podgraf označujeme jako vzorek.

3.1 Motivace

Důvody pro vzorkování populace při volbách, nebo proč se k analýze krve odebírá pouze vzorek, jsou každému zřejmé. V následující části jsou popsány problémy, které nám vzorkování pomáhá řešit nad rozsáhlými síťovými daty.

- **Nedostatek paměti** - V oblasti zpracování dat a jejich analýzy se čím dál častěji vědci i firmy nacházejí v situaci, kde je pro dosažení výsledku třeba analyzovat rozsáhlé datové sady. Práce s daty v plném rozsahu však může být velmi náročná. Například na Facebooku se k 4. čtvrtletí roku 2017 připisuje více než 2.1 miliardy měsíčně aktivních uživatelů. Takové číslo znamená, že jen uložení této datové sady bude značně náročné na použitý hardware. Množství paměti pro uložení grafu je ještě vyšší v případě, že bychom chtěli uchovávat vývoj sítě nebo obrazy v určitém čase [10].

- **Přístup k datům** - Pokud by přeci jen bylo možné pohodlně uložit data neomezené velikosti, můžeme se dostat do problémů s přístupem k těmto datům. Přesto že většina sociálních sítí nabízí API pro získání dat o uživatelích, i API mají svá omezení a není možné získat informace o všech uživatelích [22].

- **Výkon** - V rámci analýzy síťových dat se často provádějí operace, které jsou značně časově náročné. Například hledání komunitních struktur nebo průměrné vzdálenosti. Hledání průměrné vzdálenosti má pro neohodnocený graf asymptotickou časovou složitost $O(|N| * (|N| + |E|))$. Pro orientovaný graf je to $O(|N|^3)$ pomocí Floydova algoritmu. Vytvořením relevantního vzorku můžeme časovou náročnost značně snížit a tím docílit nepřímého analyzování originální sítě [20, 19].

- **Vizualizace** - K dispozici máme řadu nástrojů pro vizualizaci dat. Pro osobní počítač je to například Gephi nebo Pajek. Tyto nástroje mají však své limity, první z nich spočívá ve výkonu, který je schopen poskytnout osobní počítač, druhým problémem je čitelnost. Při zobrazení vrcholů v řádech tisíců se i s malou hustotou stane výsledná vizualizace nepřehlednou. Existují nástroje jako je například Polinode, který tyto limity posouvají k

50 000 vrcholů a 250 000 hran [31]. Za účelem vizualizace je naším cílem vytvořit vzorek splňující tyto limity.

3.2 Typy vzorkování

Vzorkování nad síťovými daty může být prováděno za různými účely, jejich společným výsledkem je graf s menším počtem vrcholů a/nebo hran než originální vzorkovaná sada. Naším cílem je, aby výsledný vzorek byl co nejblíže očekávaným vlastnostem.

3.2.1 Scale-Down

Jedná se o metodu, která má za cíl vytvořit vzorek tak, aby byl co nejpodobnější originální datové sadě a přitom se nám podařilo redukovat velikost co nejvíce. Tuto metodu můžeme formálněji definovat jako: máme graf G, který má n vrcholů a požadujeme velikost vzorku n'. Cílem je vytvořit vzorek S, mající právě n' vrcholů, který je co nejpodobnější grafu G [5]. Jinými slovy chceme, aby S zachovával vlastnosti grafu G jako jsou distribuce stupňů, centrality a další vlastnosti uvedené v 2.3.

3.2.2 Back-in-time

Jak již název napovídá, jedná se svým způsobem o cestování v čase neboli naším cílem je napodobit stav sítě v čase t. V případě, že máme kompletní temporální síť, včetně celé její evoluce, rozumným přístupem by bylo odstranění změn (přidání/odebrání hran/vrcholů) podle časového razítka nebo jejich stáří. V tomto případě máme ale k dispozici pouze aktuální stav sítě a žádné informace o stáří hran/vrcholů nebo průběhu evoluce. Jedinou informací je, že víme kolik vrcholů obsahovala síť v určitém časovém okamžiku. Následně můžeme definovat back in time sampling jako:

Mějme graf $G_{n'}$, který označuje graf G v určitém čase t, kdy měl právě n' vrcholů. Nyní chceme vytvořit vzorek S velikosti n', jehož vlastnosti se blíží $G_{n'}$ [5].

V rámci experimentů využíváme další pojmy nad časovou osou. Počáteční bod na časové ose označujeme t_0. Koncový bod na časové ose je označován t_{max}. Časový okamžik v rozsahu t_0 až t_{max} označujeme jako t_i. Graf v čase t_{max} je nazýván plným grafem (FullGraph) a označuje poslední zaznamenaný stav sítě. Plný graf je vstupem pro vzorkovací algoritmy. Graf v čase t_i označujeme jako G_0. Graf G_0 je grafem jehož vlastnosti se snažíme vzorkováním napodobit a jehož počet vrcholů je vstupem pro vzorkovací algoritmy.

3.3 Metody vzorkování

Existuje více způsobů jak členit vzorkovací algoritmy. Jedním z nich je rozdělení dle omezení přístupu k sítu. Pokud máme k dispozici celou síť můžeme využít vzorkovací metody založené na náhodném výběru. V případě s omezeného přístupu musíme využít některého z algoritmů, který vychází z prohledávání grafu. A. Clauset [12] tyto skupiny nazývá jako:

- pravděpodobnostní,
- seed-based,
- ostatní.

Mezi pravděpodobnostní řadí algoritmy, které s určitou pravděpodobností p vybírají hrany nebo vrcholy z grafu. Seed-based jsou algoritmy, které vyžadují přístup k tzv. seedům, neboli jeden nebo množinu vrcholů, které budou počátečním bodem algoritmu. Pro tuto práci použijeme rozdělení podle základního principu fungovaní daného algoritmu, tím získáme 4 skupiny:

- náhodný výběr vrcholů,
- náhodný výběr hran,
- procházení grafem,
- ostatní.

Každá z těchto skupin obsahuje základní algoritmus a jeho modifikace či rozšíření. Pro účely této práce bylo zvoleno zaměření na metody ze skupiny random walk, které v práci [5] dosahovaly nejlepších výsledkům.

3.3.1 Náhodný výběr vrcholů - Random Node

Základní metoda z této skupiny se nazývá Random Node a je založena na náhodném výběru vrcholů. Vrchol bude vybrán s pravděpodobností p, která je definována jako $p = \frac{s}{|V|}$, kde s je požadovaná velikost vzorku a $|V|$ je počet vrcholů v původním grafu. RN moc dobře nezachovává distribuci stupňů a vyžaduje přístup k celé vzorkované síti [25].

Algorithm 1 Random Node algorithm

```
1: procedure RN(n_s, G)                ▷ n_s required size of sample, G original network
2:     V_s ← ∅, E_s ← ∅
3:     while |Vs| < (n_s) do
4:         v ← RandomNode(G.Nodes)
5:         V_s ← V_s ∪ {v}
6:     end while
7:     for each e in G.Edges do
8:         (u, v) ← e
9:         if V_s.Containts(u) and V_s.Containts(v) then
10:            E_s ← E_s ∪ {(u, v)}
11:        end if
12:    end for
13:    return S(V_s, E_s)
14: end procedure
```

3.3.2 Náhodný výběr hran - Random Edge

Základní metoda z této skupiny se nazývá Random Edge a je založena na náhodném výběru hran. Hrany vybírá s uniformní pravděpodobností a přidává je do vzorku dokud není dosaženo požadované velikosti. RE narozdíl od RN zachovává relativní četnost četnost hran a zachovává distribuci stupňů. Výběr hrany připojené k vrcholu i je závislý na jeho stupni k. V případě, že v původním grafu má vrchol i stupeň k, ve výsledném vzorku bude mít vrchol stupeň $p - k$, kde p je pravděpodobnost výběru jedné hrany [25].

Počet hran ve vzorku je dán vztahem $p * |E|$, kde p je pravděpodobnost výběru jedné hrany a $|E|$ je počet hran v původním grafu. To vede k nízkému průměrnému stupni a v krajním případě, kdy průměrný stupeň klesne pod hodnotu 1 dochází k roztříštění grafu na velké množství malých komponent a ztrátě komunitních struktur [25].

Algorithm 2 Random Edge algorithm

```
1: procedure RE(n_s, G)                ▷ n_s required size of sample, G original network
2:     V_s ← ∅, E_s ← ∅
3:     while |Vs| < (n_s) do
4:         (u, v) ← RandomEdge(G.Edges)
5:         E_s ← E_s ∪ {(u, v)}
6:         V_s ← V_s ∪ {u}
7:         V_s ← V_s ∪ {v}
8:     end while
9:     return S(V_s, E_s)
10: end procedure
```

3.3.3 Náhodné procházení grafem

- **Random walk** - Tato metoda je nejjednodušším zástupcem skupiny založené na procházení grafem. RW neboli náhodná procházka je algoritmus, který s uniformní pravděpodobností vybere jeden vrchol s nenulovým stupněm, označen v_0, ze kterého je simulována náhodná procházka [25].

  Náhodná procházka znamená, že v každém kroku k se vybere jeden vrchol u z množiny sousedů vrcholu v_{k-1}. Následně vybraný vrchol u označíme jako v_k a do vzorku vložíme hranu mezi v_{k-1} a v_k. Poté iterativně pokračujeme dokud vzorek nemá požadovanou velikost. V každé iteraci probíhá rozhodování zda algoritmus bude pokračovat v náhodné procházce nebo se vrátí do původního v_0 a začne znovu. Tomuto rozšíření se někdy říká Random Walk with Reset. Pravděpodobnost návratu na začátek je označována konstantou c, jejíž doporučená hodnota je $c = 0.15$ [5]. Výsledný vzorek je vždy pouze jedna souvislá komponenta, což způsobí problémy v případě, kdy požadujeme velikost vzorku rovnu x, ale zdrojová síť obsahuje více komponent, ale ne všechny mají potřebný počet vrcholů x. Z výše uvedené vlastnosti vyplývá, že RW nezachovává distribuci komponent. RW zachovává tvar distribuce stupňů [25].

  Algoritmus může na svém začátku vybrat jako vrchol v_0, jenž se nachází v malé izolované komponentě. Pro tento případ můžeme modifikovat algoritmus, aby v prvním kroku vybral vrchol z největší komponenty. Stále může nastat situace, kde velikost největší komponenty je nižší než x, v takovém případě je vhodnější využít jinou metodu například RJ.

Algorithm 3 Random Walk algorithm

```
1: procedure RW(n_s, G, c = 0.15)        ▷ n_s required size of sample, G original network
2:     V_s ← ∅, E_s ← ∅
3:     v_0 ← RandomNode(G.Nodes)
4:     v_c ← v_0
5:     while |Vs| < (n_s) do
6:         if random(0, 1) > c then
7:             n ← Neighbours(v_c, G)
8:             u ← RandomNode(n)
9:             V_s ← V_s ∪ {u}
10:            E_s ← E_s ∪ {(u, v_c)}
11:            v_c ← u
12:        else
13:            v_c ← v_0
14:        end if
15:    end while
16:    return S(V_s, E_s)
17: end procedure
```

- **Random Jump** algoritmus vychází z metody RW. V každé iteraci probíhá rozhodování zda algoritmus vybere jednoho ze sousedů nebo bude restartován. RJ se nevrací do vrcholu v_0, který byl náhodně vybrán na začátku, ale náhodně vybírá nový vrchol. Díky tomu u RJ nedochází k zacyklení nebo uváznutí v malých komponentách. RJ nezachovává distribuci stupňů. RJ zachovává distribuci shlukovacích koeficientů [28].

- **Metropolis Haistings Random Walk** je komplexní algoritmus typu Markov chain Monte Carlo (MCMC), který nám umožnuje eliminovat prioritizování vrcholů s vyšším stupněm a tím nám zajistit zachování distribuce stupňů [29].

  Algoritmus začíná výběrem náhodného vrcholu s nenulovým stupněm, tento vrchol můžeme označit jako v. Následně je náhodně vybrán jeden ze sousedů vrcholu v. Tohoto souseda si můžeme označit jako u. V dalším kroku se u stává kandidátem. Pro přijetí vrcholu u do vzorku je ověřeno akceptační kritérium 12, kde k_u je stupeň vrcholu u, k_v stupeň vrcholu v a r náhodně vygenerované číslo z normálního rozdělení. V případě splnění podmínky akceptačního kritéria je kandidát přijat a je přidán do vzorku. Následně se vrchol stává v a algoritmus pokračuje od začátku dokud nedosáhne vzorek požadované velikosti. V případě zamítnutí akceptačního kritéria algoritmus vybírá jiného ze sousedů vrcholu v a znovu přistupuje k ověřování akceptačního kritéria.

$$r \leq \frac{k_u}{k_v} \tag{12}$$

Algorithm 4 Metropolis Haistings Random Walk algorithm

```
1: procedure MHRW(n_s, G)                    ▷ n_s required size of sample, G original network
2:     V_s ← ∅, E_s ← ∅
3:     u ← randomNode(V)
4:     while |Vs| < (n_s) do
5:         K_u ← neighbours(u)
6:         v ← randomNode(K_u)
7:         r ← random(0, 1)
8:         if r ≤ k_u / k_v then                 ▷ acceptance criterion
9:             V_s ← V_s ∪ {u}
10:            E_s ← E_s ∪ {(u, v)}
11:            u ← v
12:        end if
13:    end while
14:    return S(V_s, E_s)
15: end procedure
```

- **Forest Fire**

  Forest Fire je vzorkovací algoritmus založený na stejnojmenném modelu, jenž slouží ke generování grafů. Algoritmus pro vzorkování nad neorientovaným grafem využívá parametru dopředné pravděpodobnosti zapálení hrany (forward burning probability), označujeme jako pf. Tento parametr označuje pravděpodobnost zapálení vstupních hran vrcholu. Pro výpočet počtu sousedů k zapálení se využívá náhodného čísla z geometrického rozdělení $K \sim Geom(p)$ s průměrem $\overline{\mathrm{x}}$ podle rovnice:

  $$\overline{\mathrm{x}} = \frac{p_f}{1 - p_f} \tag{13}$$

  Algoritmus vybere s uniformní pravděpodobností vrchol u, který přidá do seznam $Burning$. Tento vrchol považujeme za zapálený. Následně pro všechny vrcholy v seznamu $Burning$ iterativně získáme seznam jejich sousedů. Podle náhodné hodnoty z geometrického rozdělení s průměrem $\overline{\mathrm{x}}$ určíme kolik sousedním vrcholů zapálíme. Zapálené vrcholy přidáme do vzorku a do seznamu $NewBurning$. Průběžně kontrolujeme, zda již není splněna podmínka pro požadovanou velikost vzorku. Po dokončení iterací nad seznamem $Burning$ nahradíme jeho obsah seznamem $NewBurning$, jenž obsahuje všechny nově zapálené vrcholy. Pokračujeme iterativně dokud nedosáhneme požadované velikosti vzorku.

  FF lze implementovat iterativně nebo rekurzivně, s využitím fronty nebo seznamu. Pseudokód 5 popisuje konkrétní implementaci použitou v rámci aplikace.

Pravděpodobnost dopředného zapálení pf jsme stanovili na hodnotu 0.2 jakožto nejefektivnější pro back in time vzorkování dle výsledků J.Leskovce. Pro scale down vzorkování je doporučená hodnota 0.7 [16].

Algorithm 5 Forest Fire

```
1: procedure FF(n_s, G, pf)          ▷ n_s required size of sample, G original network, pf
2:     V_s ← ∅, E_s ← ∅
3:     u ← randomNode(G.Nodes)
4:     Burning ← Burning ∪ {u}
5:     while |Vs| < (n_s) do
6:         Burning ← Burning ∪ {u}
7:         NewBurning ← ∅
8:         for each n in Burning do
9:             ncount ← Geometric(pf)
10:            for each u in neighbours(n) do
11:                if ncount ≤ 0 then
12:                    Break
13:                end if
14:                if |Vs| < (n_s) then
15:                    V_s ← V_s ∪ {u}
16:                    E_s ← E_s ∪ {(u, v)}
17:                    NewBurning ← NewBurning ∪ {u}
18:                    ncount ← ncount − 1
19:                end if
20:            end for
21:        end for
22:        Burning ← NewBurning
23:    end while
24:    return S(V_s, E_s)
25: end procedure
```

3.3.4 Další přístupy

Jak je uvedeno v publikaci [12] existují zcela odlišné přístupy mimo náhodný výběr nebo procházku, tím je například výběr na základě velikosti stupně. Metodu nazýváme **Top Degree**. Tato metoda vybírá n vrcholů z množiny vrcholů seřazených podle velikosti stupně. Následně mezi takto vybrané vrcholy vkládá hrany v případě, že se vyskytovaly v původním grafu.

Tato metoda zvýhodňuje vrcholy s vysokým stupněm, což může značně zkreslovat výsledky analýzy. Metoda potřebuje přístup k celé síti a vyžaduje seřazenou množinu což může ovlivnit

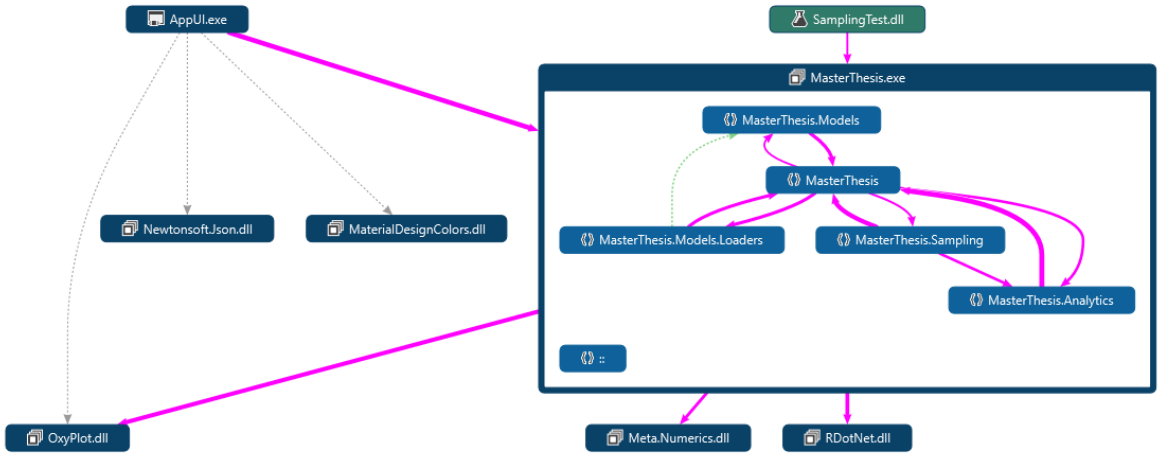
časovou náročnost algoritmu. Vhodné využití je například pro potřeby vizualizace v případě, že vrcholy s vyšším stupněm mají větší prioritu nebo význam pro síť.

4 Implementace

V této části bude popsána aplikace z pohledu návrhu, technologií a funkčnosti.

4.1 Návrh

Samotná aplikace je rozdělena na dva projekty. AppUI se stará především o grafické rozhraní. NetworkSampling je projekt nesoucí potřebné modelové třídy, například samotnou reprezentaci grafu nebo jednotlivé vzorkovací algoritmy. Na obrázku 9 je znázorněna struktura aplikace pomocí nástroje Codemap, jenž je obsažen ve Visual Studiu[1], kde nahradil původní reprezentaci návrhu pomocí UML. Růžové šipky v diagramu znárňují volání mezi aplikacemi případně jmennými prostory.

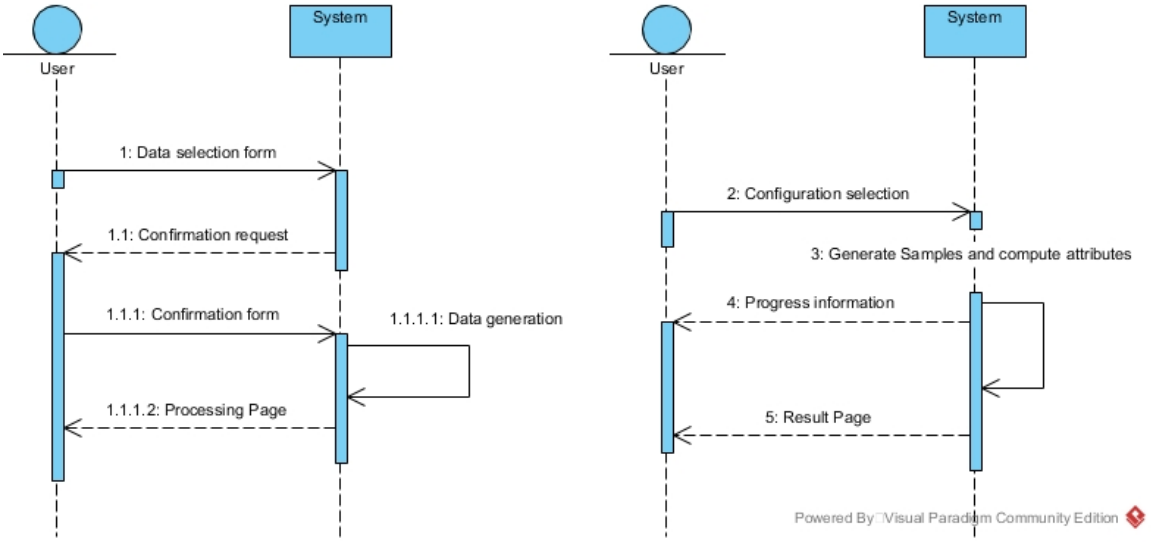


Obrázek 9: Visual Studio Code Map

Analytická část aplikace je rozdělena na části, které se starají o základní vlastnosti, komponenty souvislosti, vlastnosti vrcholů, distribuce a vlastnosti, které je potřeba získat pomocí prostředí R.

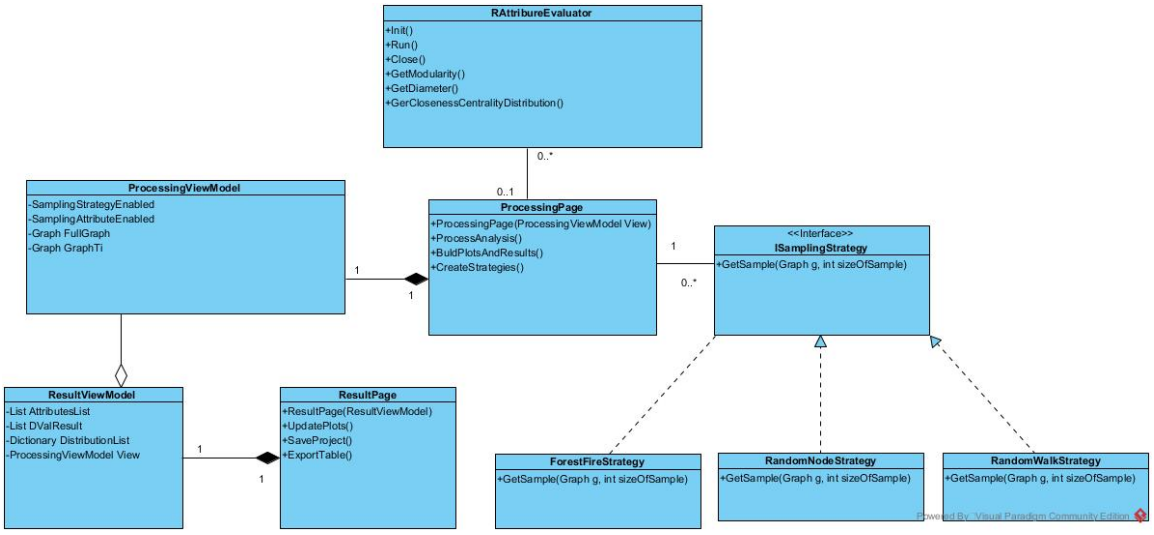
Na obrázku 10 je sekvenční diagram znázorňující interakci na úrovni systému mezi uživatelem a aplikací(System).

[1] https://www.visualstudio.com/



Obrázek 10: Sekvenční diagram - průběh práce s aplikací

Na obrázku 11 je třídním diagram znázorňující statickou strukturu aplikace. Diagram je situován tak aby znázorňoval vztahy k třídě ProcessingPage. Tato třída je obsažena v UI aplikace a obstarává logiku chování aplikace ve druhém kroku.



Obrázek 11: Třídní diagram - náhled na statickou strukturu aplikace

4.1.1 Třídy

4.1.1.1 Graf - V rámci aplikace bylo třeba vytvořit datovou strukturu, která by vhodně reprezentovala graf. Jako nejvhodnější se ukázala generická datová struktura *Dictionary* <>, jenž je ve třídě Graf použita jako $Dictionary < int, List < int >>$. Graf je do slovníku uložen tak, že jako klíč je id vrcholu a hodnotu tvoří seznam id jeho sousedů. Vrcholy samotné jsou reprezentovány celočíselným datovým typem integer. Třída graf dále poskytuje základní metody

pro práci s grafem jako jsou přidávání vrcholů nebo hran, získávání informací o sousedech daného vrcholu a další.

4.1.1.2 ISamplingStrategies je rozhraní(interface), které obsahuje pouze metodu *GetSample* (viz ukázka zdrojového kódu 1). Toto rozhraní implementují všechny vzorkovací algoritmy. Díky tomu je možné se vzorkovacími algoritmy, respektive objekty které tyto algoritmy obsahují, pracovat iterativně v poli nebo seznamu. Algoritmy vyžadující dodatečné parametry v metodě *GetSample* volají přetíženou metodu s dodatečnými parametry, které mají nastavenou výchozí hodnotu. V rámci aplikace není možnost jak tyto parametry ovlivnit. Výchozí hodnoty byly nastaveny podle výsledků experimentu 5.4.9 a výsledků v prací J.Leskovce [16].

```
public interface ISamplingStrategy
{
   Graph GetSample(Graph Source, int SizeOfSample);
}
```

Výpis 1: Rozhraní ISamplingStrategy

4.1.1.3 RAttributeEvaluator implementuje komunikaci s prostředím R pomocí knihovny **RDotNet** [1]. R je programovací jazyk a prostředí pro statistickou analýzu a grafickou reprezentaci dat. Třída obsahuje metodu Run, ve které se vytvoří instance třídy REngine zajišťující vykonávání skriptů jazyka R a načte knihovna *igraph*, v tomto okamžiku je prostředí připraveno pro zahájená práce s daty. Začne převedením objektu typu Graf do formátu, který bude vhodný pro zpracování v prostředí R. Jako nejvhodnější byla vybrána metoda seznamu hran. Po předání seznamu hran do prostředí R je vytvořen graf a ten je následně analyzován. Pomocí R a Igraph získáváme informace o asortativitě, modularitě, průměru, průměrné vzdálenosti a centralitách.

4.2 Technologie

Aplikace je vytvořena v jazyce C# za pomocí framewroku WPF. Tento framework byl zvolen z důvodu široké podpory vývojářskou komunitou i samotným Microsoftem. Další použité technologie, nástroje a doplňky:

1. **MaterialDesignInXamlToolkit**[2] je nugget balíček, který modifikuje styly nativních WPF UI komponent a přidává nové komponenty. Styly komponent odpovídají standardům materiálního designu, který definovala společnost Google.

2. **Oxyplot**[3] je opensource multiplatformní .NET knihovna pro vykreslování grafů.

3. **Accord.NET**[4] poskytuje statistickou analýzu, strojové učení, zpracování obrazu a další

[1] https://github.com/jmp75/rdotnet
[2] https://github.com/ButchersBoy/MaterialDesignInXamlToolkit
[3] http://www.oxyplot.org/
[4] https://github.com/accord-net

nástroje pro .NET aplikace. V aplikaci použit balíček Accord.Statistics k provádění statistických testů.

4. **Meta.Numerics**[5] matematická a statistická knihovna umožnující vědecké výpočty na platformě .NET.

5. **igraph**[6] - balíček analytických nástrojů do prostředí R umožňující snadnou práci s grafy a jejich analýzu.

6. **R.NET** - Jedná se o nugget balíček, který obsahuje knihovnu umožňující komunikaci s prostředím analytického nástroje R. Využívána především pro výpočet složitých operací, které jsou v R knohovně iGraph implementovány a optimalizovýny.

Aplikace byla vyvíjena v prostředí Visual Studio 2017 od společnosti Microsoft a zdrojové kódy byly verzovány pomocí verzovacího systému git[8] .

4.3 Funkcionalita

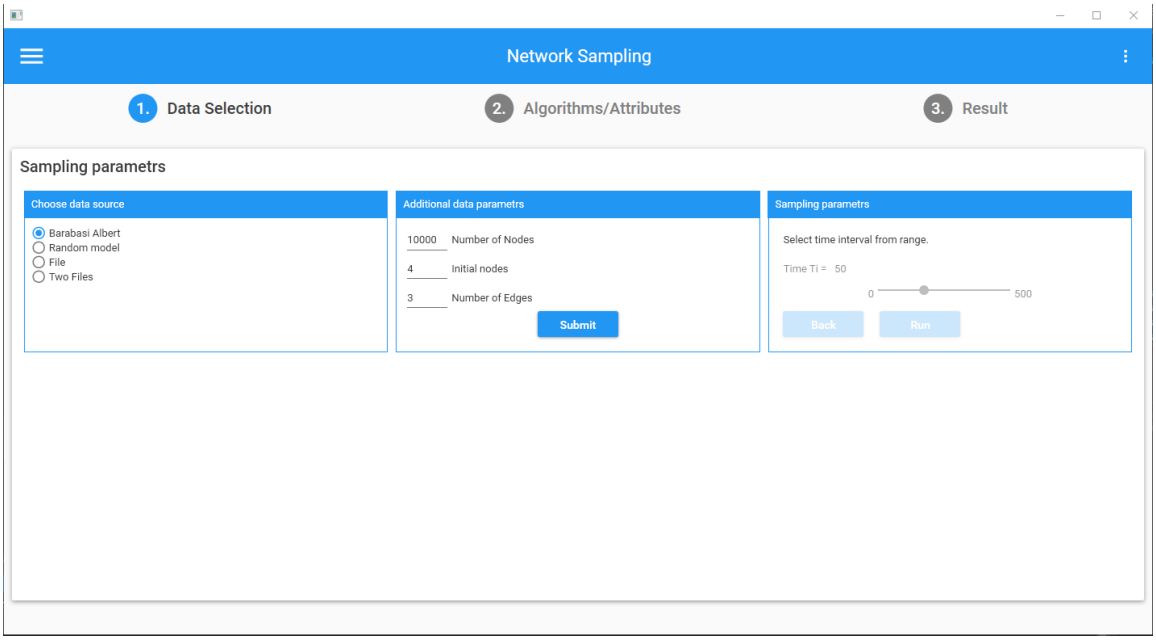
Aplikace je členěna do tří logických kroků. V této části budou popsány uživatelské možnosti v jednotlivých krocích.

4.3.1 Volba zdroje

V tomto kroku může uživatel zvolit zdroj dat pro následující analýzu. První možností je volba modelu Barabasi Albert nebo Erdos Renyi, po volbě následuje zadání parametrů dle potřeb daného modelu. Druhou možností je načtení dat ze souboru ve formátu *.csv, který obsahuje data $timestamp; id_a; id_b$. Po načtení dat nebo vygenerování sítě pomocí modelu je uživateli umožněno zvolit bod t_i na časové ose v rozsahu t_0 až t_{max}. Následně aplikace vytvoří síť v čase t_i. V případě, že síť obsahuje méně než 15% vrcholů vzhledem k síti v čase t_max, bude uživatel upozorněn na nedostatečnou velikost sítě a může zvolit pokračování nebo se vrátit na výběr časového okamžiku t_i.

Alternativně lze využít načtení dat ze dvou souborů, kde jeden je načten jako síť v čase t_i a druhý jako síť v čase t_{max}. V takovém případě jsou data očekávána ve formátu $id_a; id_b$ a soubor ve formátu *.csv nebo *.txt. Po tomto kroku uživatel nevolí časový okamžik a je pouze vyzván k potvrzení pokračování k dalšímu kroku. Snímek obrazovky aplikace v prvním kroku je zobrazen na obrázku 12.

[5] http://www.meta-numerics.net/
[6] http://igraph.org
[8] https://git-scm.com/



Obrázek 12: Snímek aplikace v prvním kroku

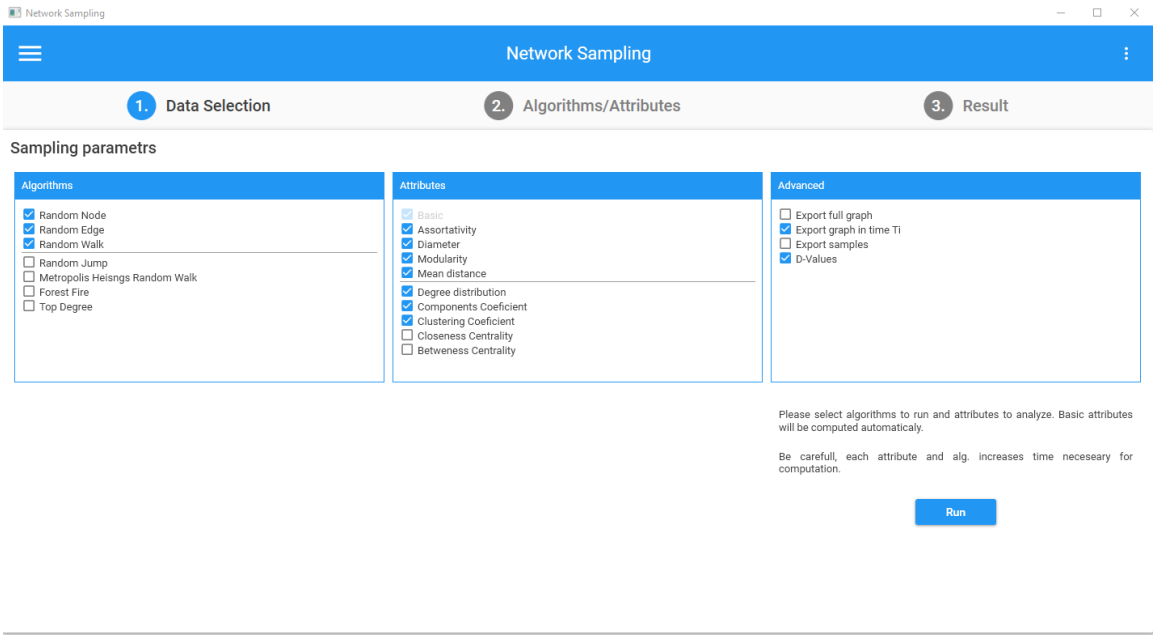
4.3.2 Volba algoritmů a vlastností

Uživatel může v tomto kroku zvolit jaké vzorkovací algoritmy bude aplikace používat, a také jaké vlastnosti se mají sledovat či vypočítávat. Důležitou poznámkou, na kterou je uživatel upozorněn, je uvážená volba vlastností k analýze, protože některé algoritmy (např. Louvain pro zjištění modularity) mají vysokou časovou složitost. Je vhodné pečlivě vybírat, které algoritmy nás opravdu zajímají.

Dále je možné zvolit, zda si uživatel přeje exportovat síť v čase t_i, t_{max} nebo jednotlivé vzorky do samostatných souborů. Exportované vzorky mohou sloužit k dodatečné analýze. Pokud exportujeme síť v čase t_i i t_{max} můžeme tyto soubory použít jako vstup pro aplikaci při dalším spuštění. Tato metoda se hodí především v případě, kdy máme jeden soubor obsahující temporální síť včetně celého jejího vývoje a chceme nad jedním časovým okamžikem t_i provádět serii experimentů. Výpočet D hodnot je dalším volitelným parametrem 5.2.1.

Po zvolení konfigurace a potvrzení bude zahájeno vzorkování a analýza. Na obrazovce se zobrazí ukazatel průběhu vzorkování a analýzy. Ukazatel obsahuje x dílků odpovídajících počtu zvolených vzorkovacích metod.

Snímek obrazovky v tomto kroku je na obrázku 13.

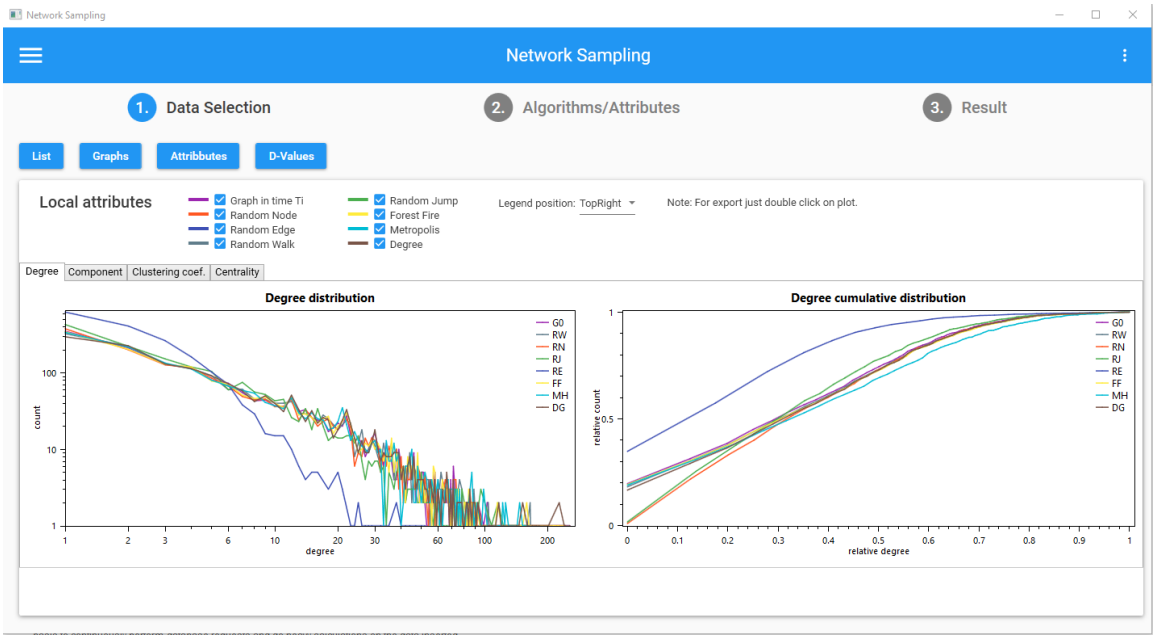


Obrázek 13: Snímek obrazovky v druhém kroku

4.3.3 Zobrazení výsledků

Na této stránce s výsledky má uživatel přehlednou tabulku s vypočtenými vlastnostmi, tabulku s vypočtenými D-hodnotami a skupinu grafů znázorňující vlastnosti reprezentovány pomocí distribucí. V horní části karty s grafy si uživatel může zvolit jaké distribuce mají být vykresleny, případně na jaké pozici se má zobrazovat legenda.

Výsledné tabulky s daty je možné exportovat do souboru *.csv. Data lze přímo zkopírovat do schránky Windows nebo uložit celý projekt pro pozdější zobrazení dosažených výsledků včetně grafů. Při načtení exportovaného projektu se aplikace přepne do třetího kroku k pouhé prezentaci výsledků. Dvojklikem na graf je možné vyvolat dialogové okno a graf uložit do formátu *.png nebo *.pdf.



Obrázek 14: Snímek obrazovky v třetím kroku

4.3.4 Problémy v rámci implementace

Nad rozsáhlými daty aplikace vykazovala vysokou paměťovou náročnost a často docházelo z zachycení vyjímky *System.OutOfMemoryException*. Tato výjimka může být zapříčiněna nadměrnou velikostí alokované paměti na jeden proces (pro 32 bit přes 800 MB) nebo vysokou fragmentací virtuální paměti a tudíž nemožností alokovat blok paměti požadované velikosti. Vzhledem k opakovanému výskytu této chyby, bylo nutné upravit nastavení sestavení aplikace pouze pro 64-bitový systém.

V případě, že aplikace bude spuštěna na zařízení s nedostatečnou velikostí operační paměti, může to zásadně ovlivnit její fungování.

V rámci aplikace bylo řešeno i několik algoritmických problémů. Například metody založené na náhodné procházce měly tendenci se zacyklit v případě, že požadovaná velikost vzorku se blížila velikosti vzorkovaného grafu. Tento problém musel být řešen dodatečným kritériem pro ukončení algoritmu. V případě, že velikost největší komponenty je menší než požadovaný počet vrcholů, je požadovaný počet nastaven na 90% velikosti komponenty. V případě metody RW je počítán počet kroků algoritmu a nastaven horní limit po jehož dosažení se algoritmus ukončí i bez dosažení počtu vrcholů.

Původním záměrem bylo udržovat vzorky v paměti a umožnit uživateli dodatečnou analýzu v případě, že se při prvním spuštění aplikace rozhodl některou vlastnost vynechat. Pro účely Back-in-time samplingu se požadovaná velikost vzorku může blížit velikosti vzorkované sady. Po vytvoření takových vzorků by potřebná paměť dosahovala násobků velikosti původní datové sady. Z tohoto důvodu není možné dodatečně dopočítávat vlastnosti vzorků.

5 Experimenty

Cílem experimentů je vytvořit vzorky v x časových okamžicích nad různými datovými sadami a vyhodnotit výsledky pomocí distribucí a KS-testu 5.2.1. Výsledky celého experimentu budou vyhodnocovány na základě průměrné D-hodnoty a distribucí pro každou datovou sadu s ohledem na její kontext a charakteristiku.

5.1 Zkoumané vlastnosti

5.1.1 Stupeň vrcholu

U stupně vrcholu využijeme jeho distribuce a průměrný stupeň. Výsledkem tohoto pozorování bude závěr zda algoritmus zachovává distribuci stupňů, případně jak dobře. Měřeno D-hodnotou vzhledem k distribuci stupňů originální sady.

5.1.2 Hustota

Hustota podle [23] také označována jako hustota hran, označuje kolik se v grafu nachází hran vzhledem k maximálnímu možnému počtu. Tato definice uvažuje prosté neorientovaný graf. Hustota v neorientovaném grafu může být vyjádřena rovnicí 14.

$$D = \frac{2|E|}{|V|(|V|-1)} \tag{14}$$

Alternativní definice hustoty může být stanovena jako podíl počtu hran a vrcholů [24].

5.1.3 Shlukovací koeficient

Vlastnost je analyzována ve formě distribuce shlukovacích koeficientů. Pro každý stupeň vrcholu je vypočtena průměrná hodnota shlukovacícho koeficientu vrcholů tohoto stupně.

5.1.4 Komponenty souvislosti

Za komponentu souvislosti považujeme souvislý podgraf, který je izolován od dalších vrcholů v grafu. Neboli jedná se o podgraf mezi jehož libovolnými dvěmi vrcholy existuje cesta a není spojen s žádnými dalšími vrcholy v grafu.

Pro vyhodnocení experimentů vytváříme distribuci souvislých komponent, kde se zaměřujeme na jejich velikost a počet. U algoritmů sledujeme zda zachovávají distribuci komponent. Na ose X bude velikost komponenty, na ose Y počet takových komponent. V tabulce atributů je k dispozici celkový počet komponent v síti.

V krajním případě že vzorek má 1 komponentu a síť v čase t_i více než 3 komponenty je automaticky dosazena D hodnota 1. Pokud má vzorek více než 3 komponenty a síť v čase t_i méně než 3, D hodnota je 1. V případě, že vzorek i síť v t_i mají měně než 3 komponenty je D

hodnota 0. Tyto nastavení zajišťují případy, kdy nemáme dostatek bodů pro vytvoření distribuce pro spuštění KS testu.

5.1.5 Assortativita

Assortativita vyjadřuje míru s jakou vrchol preferuje připojování k vrcholům, které jsou v určitém ohledu podobné. Z pravidla se setkáváme s podobností na základě velikosti stupně vrcholu. Assortativita může nabývat hodnot v rozsahu <-1,1>. V případě, že se hodnota assortativity blíží +1 znamená to, že vrcholy v síti preferují připojování k vrcholům stejného stupně. V případě opačném, tedy hodnota blížící se -1, vrcholy preferují připojování v vrcholům odlišného stupně.

Assortativita je významným parametrem při studiích epidemií, kde nám pomáhají porozumět šíření nemoci nebo léku.

5.1.6 Modularita

Modularita je jedna z vlastností vztahující se ke struktuře sítí. Modularita byla navržena tak, aby měřila sílu rozdělení sítě na komunity. Síť s vysokou modularitou bude vykazovat husté propojení vrcholů v rámci komunit a nízký počet vazeb mezi vrcholy mimo komunitu. Vysokou modularitu vykazuje například síť interakce proteinů nebo jiné biologicky inspirované sítě [30].

$$e_{ii} = |\{(u, v) : u \in V_i, v \in V_i, (u, v) \in E\}|/|E| \tag{15}$$

$$a_i = |\{(u, v) : u \in V_i, (u, v) \in E\}|/|E| \tag{16}$$

V rovnici 15 e_{ii} vyjadřuje podíl hran v rámci komunity i, kde u a v jsou vrcholy náležící mezi množinu vrcholů komunity V_i a společně tvoří hranu (u, v), E je množina hran a $|E|$ je počet hran. V rovnici 16 e_{ii} vyjadřuje podíl hran s alespoň jedním vrcholem v komunitě i.

$$Q = \sum_{i=1}^{k} (e_{ii} - a_i^2) \tag{17}$$

Výsledná modularita Q je vyjádřena jako suma rozdílů mezi pravděpodobností existence hrany v rámci komunity a pravděpodobností existence náhodné hrany v komunitě i na druhou, pro všechny moduly v rámci sítě [33].

V rámci experimentů budeme pozorovat zda výsledný vzorek zachovává hodnotu modularity.

5.1.7 Betweenness centralita

Betweenness je míra která vychází z předpokladu, že čím více je vrchol důležitější, tím větší počet nejkratších cest přes tento vrchol povede. Formálně lze zapsat jako 18 kde $\sigma_{st}(v)$ je počet

nejkratších cest procházejících vcholem v a σ_{st} je počet všech nejkratších cest mezi vrcholy s a t.

$$g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{18}$$

Tento typ centrality hraje významnou roli například v komunikačních sítích, kde znázorňuje významnost jednoho uzlu vzhledem k tomu kolik spojení přes něj vede.

5.1.8 Closeness centralita

Tato míra je vypočítána jako suma nejkratších cest mezi vrcholem a všemi ostatními vrcholy v grafu. Vyplývá z předpokladu, že čím blíž je vrchol ke všem ostatním vrcholům, tím více je důležitý. V rovnici 19 je vyjádřena closeness centralita vrcholu u, kde $d(u, i)$ je vzdálenost mezi vrcholy u a i.

$$C(u) = \frac{1}{\sum_i d(i, u)} \tag{19}$$

5.1.9 Průměr v čase

Naším cílem je ověřit jak je průměr zachováván ve vztahu k času, jinými slovy budeme sledovat jak se vyvíjí průměr sítě se zvyšujícím se počtem vrcholů. Výsledek pro jednu datovou sadu a jednu metodu získáme ve formě distribuce a tu následně porovnáme mezi jednotlivými metodami.

5.2 Ověřovací technika

Vzorky, které získáme pomocí vzorkovacích algoritmů musíme podrobit analýze a získat informace o jejich vlastnostech, ať už se jedná o jednotlivé číselné hodnoty nebo celé distribuce. Z těchto zjištěných údajů potřebujeme vyhodnotit závěr o míře podobnosti vzorku k původní síti v čase t_i. Porovnání globálních vlastností je poměrně jednoduché díky přehledné tabulce. Problémem zůstává jak porovnávat distribuce. K tomuto účelu využijeme statistického testu Kolmogorov-Smirnov. Tento test nám umožňuje porovnat každou distribuci originální sítě s distribucí vzorku.

5.2.1 Kolmogorov-Smirnov test

KS-Test je ověřovací technika ve statistické analýze, která umožňuje testovat zda dvě náhodné proměnné pocházejí ze stejného rozdělení pravděpodobnosti, případně zda jednorozměrná náhodná proměnná má předpokládané teoretické rozdělení. Výsledkem tohoto testu je D-hodnota, která označuje maximální absolutní rozdíl mezi aktuální a očekávanou kumulativní distribucí [27].

V rovnici 20 je vyjádřen výpočet D-hodnoty pro dvě distribuce. Předpokládejme, že máme dvě distribuce označeny d_1 a d_2, potom $F_1(x)$ je kumulativní distribuční funkce pro d_1 a $F_2(x)$ je kumulativní distribuční funkce pro d_2. sup_x označuje funkci supremum.

$$D = sup_x\{|F_1(x) - F_2(x)|\} \tag{20}$$

Výhodou KS-testu je, že nemusíme znát rozdělení dat před spuštěním testu. D-hodnota je jednoduchá na výpočet a test není omezen na velikost vstupních distribucí a lze jej použít i na malé sady.

5.3 Popis datových sad

Charakteristika datové sady a kontext v jakém datová sada vznikla je jedním z nejdůležitějších aspektů pro správnou analýzu a pochopení výsledků z analýzy vyplývajících. V této části budou popsány jednotlivé datové sady, obsah datových souborů, jejich formát, rozsah a časová složka. Dále je věnována pozornost charakteru časové složky a případné modifikaci pro potřeby analýzy.

5.3.1 Generovaný bezškálový graf

Pro generování bezškálového grafu byl použit Barabasi-Albert model. Parametry pro tento model jsme nastavili jako $n = 20000; n_0 = 100; m = 3$. Výsledný graf obsahuje 20000 vrcholů a 59900 hran.

Časová složka u tohoto grafu byla vytvořena na stupnici $0 - n$, kde n je počet vrcholů. Díky tomuto přístupu odpovídá časové razítko vzniku vrcholu a jeho hran id vrcholu.

5.3.2 Generovaný náhodný graf

Časová složka u tohoto grafu je obtížně konstruovatelná, protože dle definice 2.4.1 jsou vytvořeny nejprve vrcholy a následně jsou mezi ně vkládány hrany. To by znamenalo, že graf v čase t_0 by obsahoval stejně vrcholů jako v t_i. Toto chování jsem pro účely analýzy vyhodnotil jako nežádoucí a přistoupil k metodě uvedené u BA modelu. Tedy časově razítko v rozmezí 0 až n, kde n je počet vrcholů grafu a číslo přiřazené vrcholu odpovídá pořadí v jakém byl přidán. Pokud ve vygenerovaném náhodném grafu existuje mezi vrcholy u a v existuje hrana, bude tato hrana existovat od času t_i, ve kterém jsou již v síti obsazeny vrcholy u a v.

Parametry pro generování náhodného grafu jsou zvoleny jako počet vrcholů $n = 10000$ a počet hran $m = 130000$.

5.3.3 Kontakt na pracovišti

Temporální síť obsahuje informace o osobách a o jejich face-to-face kontaktu na pracovišti. Tato reálná data byla sesbírána ve Franci v rozsahu 7 dnů od 24. června do 3. července 2013. Tento dataset obsahuje dva soubory. První obsahuje tabulátorem separovaný seznam, kde každý řádek

má formát "t i j", kde i a j jsou unikátní anonymní id jednotlivých osob a t označuje časovou složku během které, byl kontakt mezi osobami aktivní. t je uvedeno ve vteřinách od počátku měření tj. od 00:00 24. června 2013) a označuje časový úsek 20s. V případě, že by osoby byly v kontaktu např. 60s, v souboru by existovaly 3 záznamy s časovým razítkem t, $t + 20$ a $t + 40$. Soubor obsahuje 92 aktérů, mezi nimiž bylo zaznamenáno 9827 kontaktů během 7104 časových intervalů v rozsahu 28820 až 1016440 [13].

Pro potřeby zpracování tohoto datasetu byl soubor převeden do formátu .csv a časové informace upraveny do časových oken odpovídajících jednotlivým dnům měření. Důvodem pro tuto úpravu je fakt, že v originálním souboru je v jednom okamžiku t příliš málo aktivních hran. V případě, že existuje hrana mezi vrcholy u a v v rámci jednoho dne, je tato hrana vložena do souboru. Výsledný soubor obsahuje 11 časových okamžiků.

5.3.4 Facebook like - sociální sítě

Facebook-like Social Network pochází z online community pro studenty na Univesity of California ve městě Irvine. Vrcholy v této síti jsou osoby v rámci sociální sítě podobné facebooku. Hrana mezi dvěmi vrcholy existuje v případě, že mezi dvěmi osobami proběhla přímá komunikace pomocí zprávy. Časová složka označuje datum a čas odeslání zprávy [14].

Dataset obsahuje 1,899 uživatelů, kteří celkem odeslali 59 835 zpráv což vytvořilo 20 296 spojení mezi uživateli s různým ohodnocením. Ohodnocení hrany vyjadřuje počet znaků obsažených ve zprávě. Pro účely analýzy ohodnocení hrany zanedbáváme.

Časový údaj je v tomto datasetu uveden ve formátu časového razítka "2004-04-22 02:49:55", to je pro účely analýzy převedeno do celočíselného formátu. Pro účely analýzy byla přesnost časového razítka omezena na dny, díky tomu během jednoho časového okamžiku přibývá více aktivních hran. Prvnímu jedinečnému časovému okamžiku t je přiřazena hodnota 1, ta je inkrementována s každým dalším jedinečným časovým okamžikem. V souboru se objevují hrany, kde je vrchol propojen sám se sebou. Takové hrany jsme při exportu do univerzálního formátu vynechali. Ve výsledku jsme dostali 193 časových okamžiků, tedy 193 dnů na časové ose.

5.3.5 IMDB

Databáze IMDB[1] obsahuje data o hercích a titulech na kterých pracovali. Pomocí parseru jsme vyextrahovali síť spolupráce mezi herci, režiséry, spisovateli a dalšími rolemi v oblasti filmu, kde spojení mezi dvěma aktéry je realizováno v případě, že se podíleli na tvorbě stejného titulu (filmu, seriálu, atd). Výběr vhodné datové sady pro účely experimentů závisel především na rozsahu datového souboru. Časová složka u tohoto typu sítě označuje rok v jakém byl vydán titul na kterém aktéři spolupracovali.

V případě extrakce sítě spolupráce herců došlo k vytvoření datového souboru, který svou velikost přesahoval 5GB, proto jsem přistoupil k restrikcím podle časového údaje. Síť v rozmezí

[1] https://www.imdb.com/

let 2001 až 2017 měla velikost 2GB. Tyto restrikce se ukázaly jako nedostatečné. Změnou role z herců na režiséry a restrikcí na časové období od r. 1990 do roku 2000 jsme dostali výslednou datovou sadu přijatelné velikosti cca 20MB s celkovým počtem 94196 vrcholů a 143218 hran.

V případě této datové sady se nabízejí dvě možnosti jak vnímat trvání existence hrany. První možnost spočívá ve vytvoření hrany v době první spolupráce mezi aktéry a ponechání této hrany permanentně. Taková síť pouze roste počtem vrcholů i hran, nezahrnuje však opakující se spolupráce. Alternativně lze zachovat hrany pouze v čase, ve kterém se vyskytla spolupráce aktérů. V dalším roce se již tato hrana nebude vyskytovat, pokud neproběhla další spolupráce na jiném titulu. Pro účely experimentů byl zvolen první přístup. Hrany jsou od jejich vytvoření permanentní a síť v čase pouze roste.

5.3.6 Autonomous systems

Síť cest v rámci internetu může být organizován do podsítí nazývaných autonomní systémy. Autonomní systém je typicky síť poskytovatele internetových služeb, datacentra nebo rozsáhlé korporátní sítě. AS rozdělují směrování na internetu na dvě části, směrování mezi AS a uvnitř AS. Díky logům směrovacího protokolu BGP (Border Gateway Protocol) jsme schopni zkonstruovat síť komunikace mezi AS.

Autonomous systems AS-733 je dataset, který byl vytvořen v rámci University of Oregon Route Views Project [35]. Výsledný dataset obsahuje 733 pozorování v rámci 785 dnů od 8. listopadu 1997 do 2. ledna 2017, kde stav sítě v každém dni je uložen do samostatného souboru. Síť v určitém čase obsahuje maximálně 6474 vrcholů a 13895 hran [15].

Rouzdíl oproti sítím spolupráce spočívá ve faktu, že tati síť není pouze rostoucí, tedy s každým dalším časovým okamžikem mohou hrany a vrcholy být přidávány i odebírány.

5.4 Hledání nejlepší vzorkovací metody

Pro každou datovou sadu jsme provedli serii experimentů. Experiment obsahuje zvolení časových okamžiků t_i, algoritmů a vlastností k analýze. Volba množství časových okamžiků pro analýzu záleží na konkrétní datové sadě, cílem je vždy zvolit 3 - 5 takových okamžiků.

5.4.1 Bezškálová síť

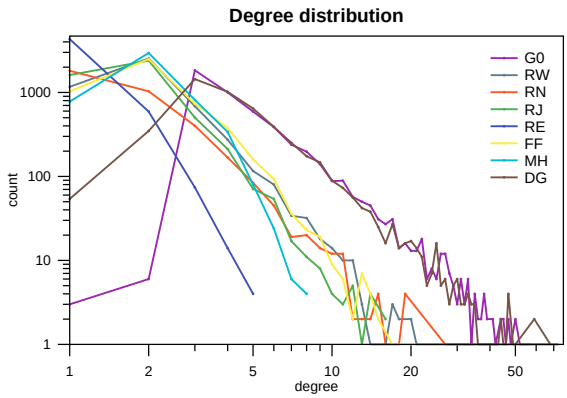
Byly zvoleny 3 časové okamžiky v rozsahu 0 až 20000. $t_1 = 5000$, $t_2 = 10000$, $t_3 = 15000$. Pro každý časový okamžik byla provedena kompletní analýza. Počet vrcholů v čase t je roven velikosti časového razítka.

V tabulce 1 jsou výsledné D-hodnoty pro bezškálovou síť vygenerovanou BA modelem a vzorek vytvořený vzhledem k času $t1$. Nejlepších výsledků dosáhly metody Random Walk a Forest Fire. Tyto metody spolu s MHRW v tomto případě zachovají distribuci komponent, jelikož i originální síť G_0 v čase t_i je tvořen jednou komponentou souvislosti.

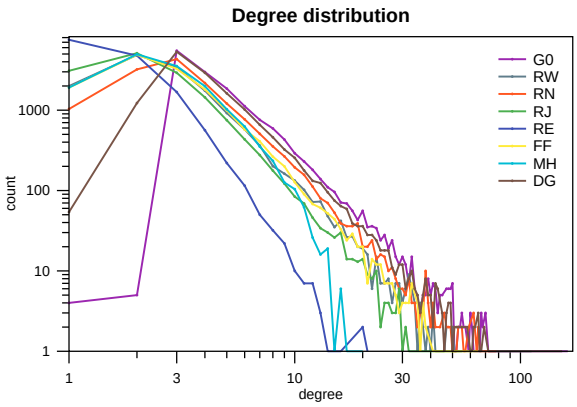
Tabulka 1: D-hodnoty nad bezškálovou sítí v čase t_1

Method	deg	clus	comp	close	betw	avg
Random Walk	0.336	0.235	0	0.177	0.362	0.222
Random Node	0.297	0.308	1	0.112	0.165	0.3764
Random Jump	0.345	1	1	0.092	0.13	0.5134
Random Edge	0.494	1	1	0.162	0.261	0.5834
Forest Fire	0.363	0.224	0	0.059	0.098	0.1488
Metropolis	0.413	1	0	0.125	0.261	0.3598
Top Degree	0.288	0.053	1	0.05	0.115	0.3012

Za zmínku stojí výsledek metody Top Degree, která vychází jako nejlepší s ohledem na parametr zachování distribuce stupňů, shlukovacího koeficientu a closenness centrality.

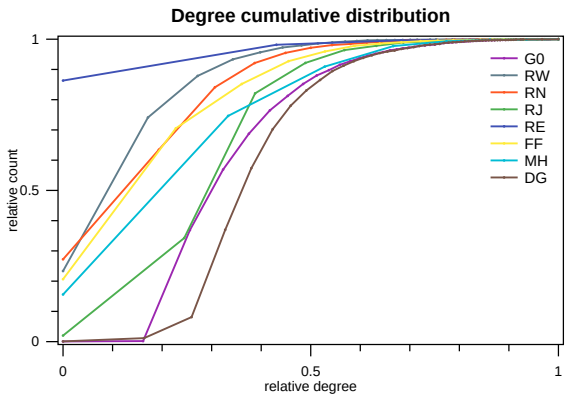


Obrázek 15: Distribuce stupňů v čase t_1

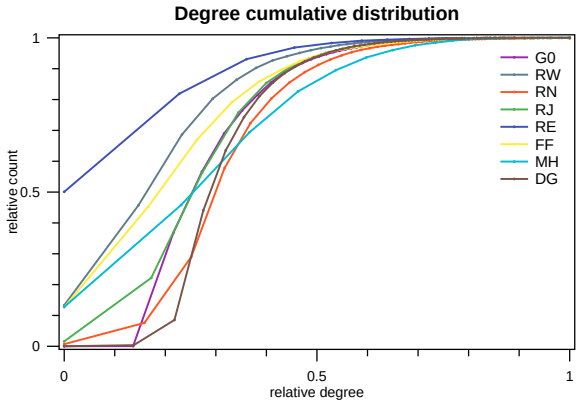


Obrázek 16: Distribuce stupňů v čase t_3

Na obrázcích 15 a 16 je viditelné srovnání distribuce stupňů v čase t_1 a t_3. Graf v čase t_i je označen jako G_0. Nad vzorkem t_3, s vyšším počtem vrcholů, jsou si distribuce jednotlivých metod mnohem více podobné. Metoda Top Degree kopíruje distribuci G_0 velice přesně. D-hodnota však porovnává distribuce na základě kumulativní distribuce viz obrázky 17 a 18.



Obrázek 17: Kumulativní diststribuce stupňů v čase t_1



Obrázek 18: Kumulativní distribuce stupňů v čase t_3

Tabulka 2: Průměrné D-hodnoty nad bezškálovou sítí v časech $t_{1,2,3}$

Method	deg	clus	comp	close	betw	avg
Forest Fire	0.279	0.1417	0	0.0543	0.122	0.1194
Random Walk	0.2713	0.1433	0	0.099	0.1943	0.1416
Metropolis	0.3763	0.437	0	0.0997	0.2227	0.2271
Top Degree	0.298	0.042	0.6667	0.039	0.109	0.2309
Random Node	0.291	0.1417	1	0.068	0.187	0.3375
Random Jump	0.2903	0.4357	1	0.069	0.2293	0.4049
Random Edge	0.5673	1	1	0.1227	0.2473	0.5875

Dle tabulky 2, kde jsou uvedeny průměrné hodnoty z časových okamžiků t_1, t_2 a t_3, vykazují metody Forest Fire a Random Walk nejlepší průměrnou D-hodnotu. Nejhorší výsledky podává metoda Random Edge, která nerespektuje distribuci velikostí komponent a shlukování.

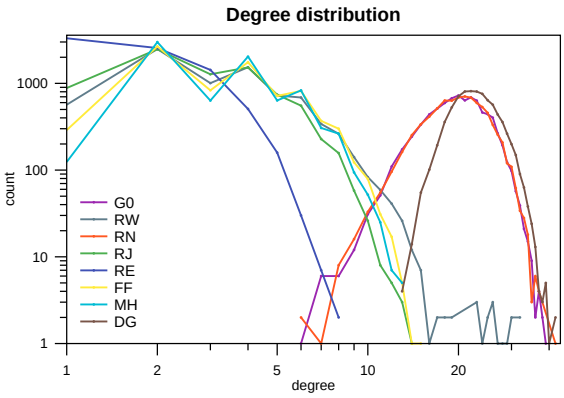
5.4.2 Náhodná síť

Náhodný graf byl vygenerován dle parametrů uvedených v 5.3.2 a byly zvoleny 4 časové okamžiky (2000, 4000, 6000, 8000). V tabulce 3 jsou uvedeny průměrné D-hodnoty ze všech 4 časových okamžiků. Z těchto dat vyplývá, že nejlépe si z pohledu D-hodnoty vedla metoda Random None následována metodami Random Edge, Forest Fire a Top Degree.

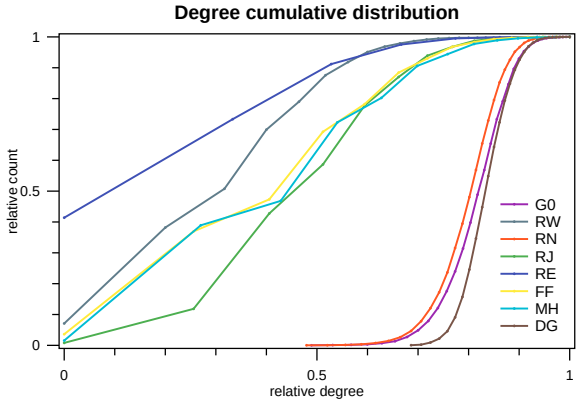
Tabulka 3: Průměrné D-hodnoty nad náhodnou sítí v časech $t_{1,2,3,4}$

Method	deg	clus	comp	close	betw	avg
Random Node	0.1135	0.061	0	0.0523	0.117	0.0688
Random Edge	0.144	0.2915	0.25	0.0683	0.1573	0.1822
Forest Fire	0.2938	0.1988	0.25	0.072	0.1528	0.1935
Top Degree	0.2755	0.1878	0.25	0.1023	0.23	0.2091
Random Jump	0.3653	0.3658	0.5	0.0788	0.1748	0.2969
Random Walk	0.2843	0.783	0.75	0.0813	0.342	0.4481
Metropolis	0.3968	0.7805	0.75	0.1168	0.205	0.4498

Metoda Random Node zachovává distribuci velikostí komponent. Metody založené na náhodné procházce mohou vykazovat D-hodnotu 0 pouze v případě, že síť v čase t_i je tvořena jednou souvislou komponentou. Random Node a Top Degree nejlépe zachovávají distribuci stupňů, což dokazují grafy na obrázku 19 a 20.

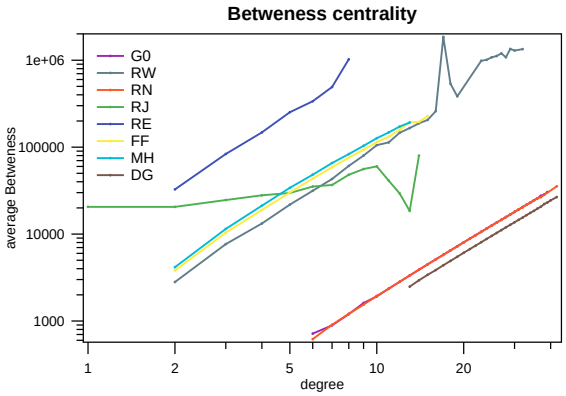


Obrázek 19: Distribuce stupňů v čase t_4

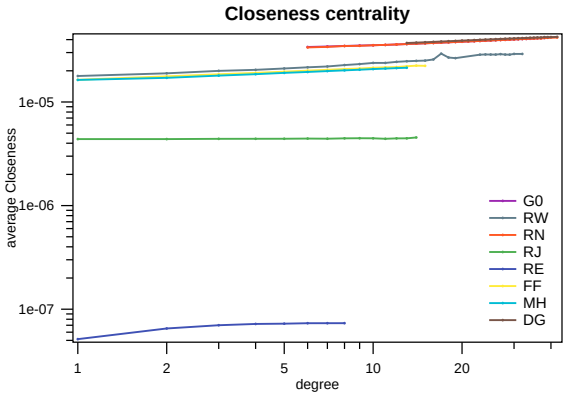


Obrázek 20: Kumulativní distribuce stupňů v čase t_4

Na obrázcích 21 a 22 jsou grafy centralit, které zobrazují, že výsledné distribuce metody Random Edge se přímo překrývají s distribucí grafu v čase t_i. Všechny metody založené na náhodné procházce nad náhodným grafem dosahují podprůměrných výsledků, nejlepší z této skupiny je metoda Forest Fire.



Obrázek 21: Betweness centralita v čase t_4



Obrázek 22: Closeness centralita v čase t_4

5.4.3 Kontakt na pracovišti

Byly zvoleny 3 časové okamžiky t_i nabývající hodnot 3,6 a 9. Časové okamžiky byly zvoleny tak, aby byly rovnoměrně rozloženy na časové ose. Vzhledem k velmi malé velikosti tohoto datového souboru není možné jednoznačně určit nejlepší vzorkovací metodu, toto tvrzení potvrzují i velmi malé rozdíly v průměrné D hodnotě dle tabulky 4. Výsledky opakovaného experimentu nad stejným časovým okamžikem mohou být pro každý pokus velmi odlišné, přesto metoda Random Edge nejčastěji podává podprůměrné až nejhorší výsledky.

Tabulka 4: Průměrné D-hodnoty nad sítí kontaktů na pracovišti v časech $t_{1,2,3}$

Method	deg	clus	comp	close	betw	avg
Forest Fire	0.265	0.242	0	0.257	0.617	0.276
Random Node	0.265	0.231	0	0.271	0.711	0.296
Top Degree	0.342	0.28	0	0.264	0.729	0.323
Random Walk	0.353	0.319	0	0.315	0.704	0.338
Metropolis	0.32	0.396	0	0.359	0.904	0.396
Random Edge	0.552	0.29	1	0.301	0.549	0.538
Random Jump	0.463	0.23	1	0.211	0.509	0.683

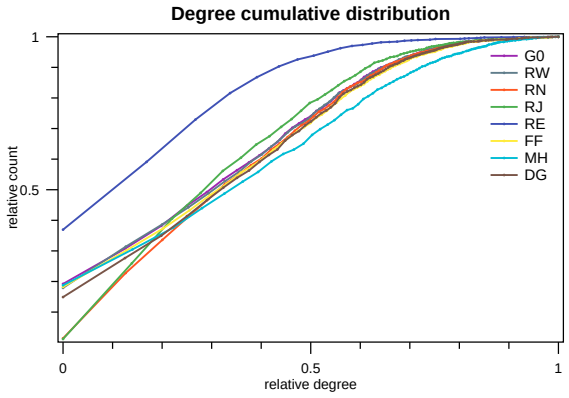
5.4.4 Facebook-like síť

S uniformní pravděpodobnostní jsme zvolili 4 časové okamžiky t_1 až t_4 v rozsahu 1 až 193, ve kterých je velikost sítě alespoň 15% vzhledem k síti v čase t_{max}. Tyto časové okamžiky nabývají hodnoty 58,93,97,122.

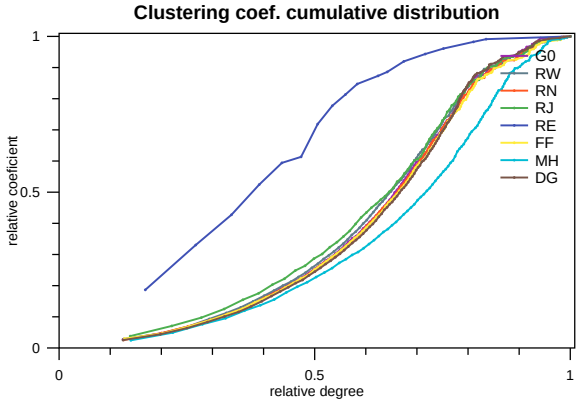
Tabulka 5: Průměrné D-hodnoty nad náhodnou Facebook-like sítí v časech $t_{1,2,3,4}$

Method	deg	clus	comp	close	betw	avg
Top Degree	0.144	0.033	0	0.032	0.076	0.057
Forest Fire	0.24	0.051	0.75	0.051	0.087	0.236
Random Walk	0.181	0.03	1	0.03	0.095	0.267
Random Jump	0.179	0.045	1	0.045	0.14	0.282
Random Node	0.181	0.035	1	0.033	0.197	0.289
Metropolis	0.277	0.087	1	0.086	0.125	0.315
Random Edge	0.234	0.128	1	0.1	0.12	0.316

U parametru zachování distribuce velikostí komponent vycházela s D-hodnotou 0 pouze metoda Top Degree. Po hloubším zkoumání se ukázalo, že v datové sadě existují řádově jednotky izolovaných komponent obsahujících 2 až 4 vrcholy. Zanedbání těchto komponent zásadně vylepšilo výsledky metod založených na náhodné procházce. V tabulce 5 průměrných D-hodnot napříč časovými okamžiky vede s nejlepší průměrnou D-hodnotou metoda Top Degree následována metodami Random Walk a ForestFire. Jako nejhorší vychází metody Metropolis Hasting a Random Edge.

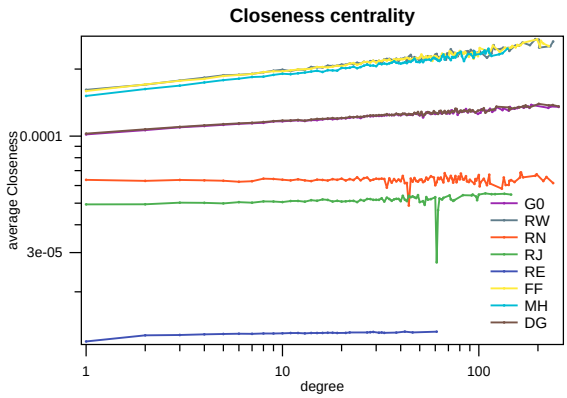


Obrázek 23: Kumulativní stupeň v čase t_4

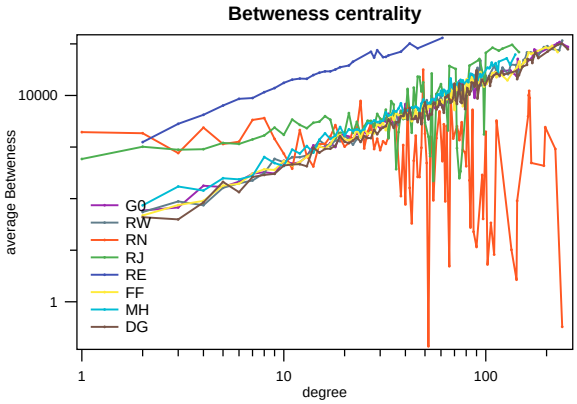


Obrázek 24: Shlukovací koef. v čase t_4

Všechny metody, s vyjímkou Random Edge, dobře napodobují distribuci stupňů (viz. obr. 23) i shlukovací koeficient (viz. obr. 24). Metoda Metropolis Hasting je druhou nejhorší, tvar distribucí však alespoň přibližně zachovává. V případě Closeness centrality, na obr. 25, se algoritmy rozdělily do tří skupin. Metody založené na náhodné procházce si vedou všechny velmi podobně. Nejlepší shody dosahuje metoda Top Degree. U betweness centrality, viz. obr. 26, zcela propadají metody Random Jump, Random Node a Ranom Edge. Naopak poněkud překvapivě vzhledem k dosavadním výsledkům si dobře v tomto parametru vede metoda Metropolis Hasting spolu s Forest Fire, Top Degree a Random Walk.



Obrázek 25: Closeness centralita v čase t_4



Obrázek 26: Betweness centralita v čase t_4

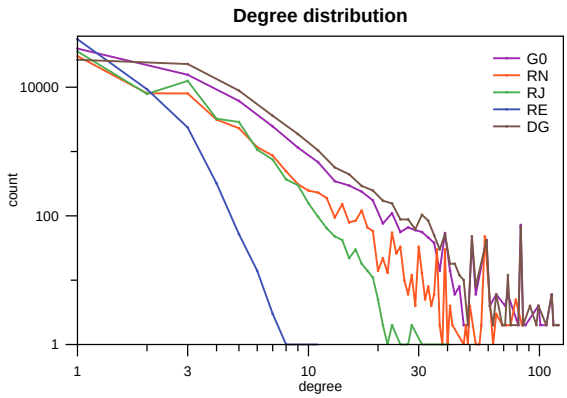
5.4.5 IMDB síť

Tato datová sada je co do počtu vrcholů a hran největší z analyzovaných. Z rozsahu let 1991 - 1999 jsme zvolili 3 časové okamžiky (1992,1994,1996). Z výsledné analýzy jsme získali informaci o velkém počtu komponent které tvoří síť, kvůli tomuto faktu jsou metody založené na náhodné procházce, vytvářející jednu souvislou komponentu, předem odsouzeny k neúspěchu.

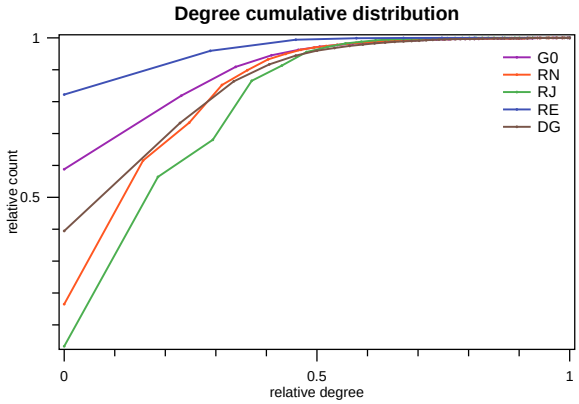
Tabulka 6: Tabulka vlastností k vzorku v čase t_3

Graph Name	nodes	edges	deg	dist	wcc	clust	modul	ass	dia
Graph in time t_i	67742	100427	2.965	8.6553	24840	0.39906	0.9896	0.889	32
Random Node	67742	73103	2.158	10.4256	31791	0.36969	0.9914	0.8436	28
Top Degree	67742	129991	3.838	10.3072	20512	0.58449	0.9871	0.8414	30
Random Jump	67742	70283	2.075	12.3734	24339	0.30083	0.9964	0.5636	34
Random Edge	67743	41519	1.226	3.5813	27983	0.0434	0.9999	0.3296	28
Random Walk	1906	13410	14.071	6.6177	1	0.67313	0.7816	0.4956	23
Metropolis	3349	15109	9.023	11.0591	1	0.61841	0.9061	0.3318	33
Forest Fire	3349	25400	15.169	8.3902	1	0.7696	0.8523	0.7146	22

Z tabulky 6 můžeme vyčíst vysokou modularitu a asortativitu. Průměrný stupeň dosahuje v tomto měření velmi vysokého rozsahu <2,15>. Vysoký průměrný stupeň vykazují především metody založené na náhodné procházce, tyto metody musí projít mnohem více cest a přidat více hran aby dosáhly požadovaného počtu vrcholů. Metody Random Walk, Metropolis Hasting a ForestFire vytvořily pouze jednu komponentu a nebyly schopny dosáhnout potřebného počtu vrcholů, algoritmy dosáhly pouze vzorku o velikosti 90% z největší komponenty ve vzorkované síti, toto kritérium bylo zvoleno pro časovou optimalizaci algoritmů. Metody RW, MHRW a FF vyloučíme z dalšího porovnávání nad touto datovou sadou.

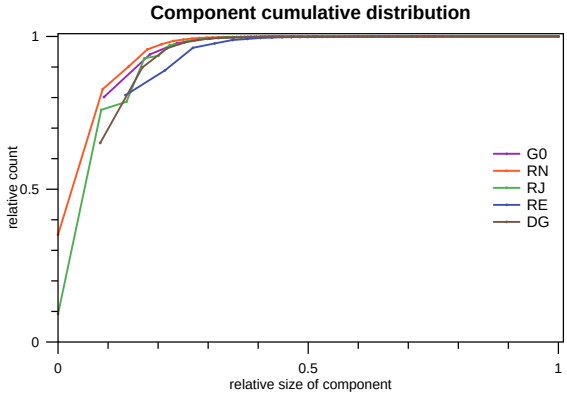


Obrázek 27: Stupeň v čase t_4

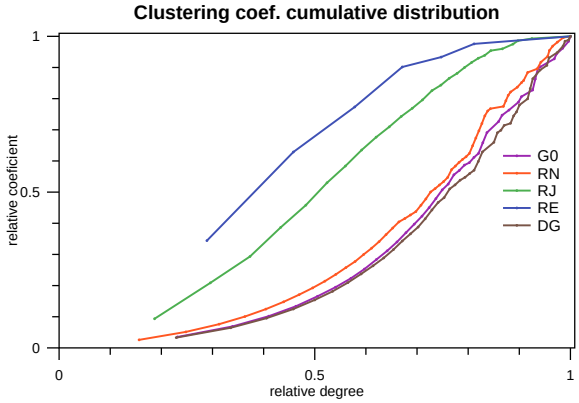


Obrázek 28: Kumul. stupeň v čase t_4

Na obrázcích 27 a 28 vidíme distribuci stupňů a kumulativní distribuci stupňů. Z obou grafů je patrný špatný výkon metody Random Edge a dobré zachování distribuce stupňů metodami Random Node a Top Degree.

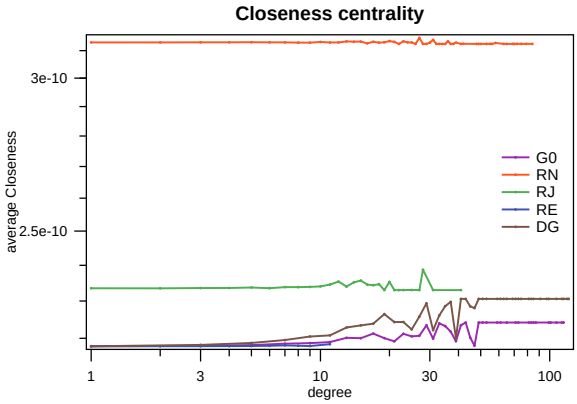


Obrázek 29: Kumul. dist. komponent v čase t_4



Obrázek 30: Shlukovací koef. v čase t_4

67742 vrcholů je v čase t_3 rozloženo mezi 24840 komponent, což je znázorněno na obrázku 29 kumulativní distribucí komponent. Největší komponenta má 3349 vrcholů. Graf je rozdělen na velké množství malých komponent, které jsou hustě propojeny což potvrzuje vysokou modularitu a má velký vliv na vysokou asortativitu pohybující se u vzorků v rozsahu <0.5,0.9> . Na obrázku 30 je distribuce shlukovacího koeficientu, kde metody Random Edge a Random Jump podávají nejhorší výsledky.



Obrázek 31: Closeness centrality v čase t_4

Distribuce closeness centrality na obrázku 31 ukazuje, že nejlepší shody dosahuje metoda Top Degree. Průměrné D hodnoty v tabulce 7 ukazují, že nejlepších výsledků dosahuje metoda Random Node a Top Degree .

Tabulka 7: Průměrné D-hodnoty nad náhodnou sítí

Method	deg	clus	comp	close	betw	avg
Random Node	0.418	0.076	0.666	0.067	0.512	0.348
Top Degree	0.417	0.083	0.825	0.083	0.447	0.371
Random Jump	0.575	0.116	0.749	0.104	0.48	0.405
Forest Fire	0.465	0.105	1	0.102	0.453	0.425
Random Walk	0.525	0.081	1	0.079	0.472	0.432
Metropolis	0.464	0.109	1	0.108	0.487	0.434
Random Edge	0.71	0.226	0.855	0.154	0.526	0.494

5.4.6 Autonomní systémy

Pro účely analýzy bylo zvoleno 10 náhodných časových okamžiků viz tabulka 8. Podle průměrné D hodnoty ze všech časových okamžiků v tabulce 9, vychází jako nejlepší metoda Random Walk následována metodami Forest Fire a Metropolis Hasting. Nejhůře dopadla metoda Random Edge. Velký vliv na výsledek má zachování distribuce komponent, v tomto případě zachování jediné komponenty.

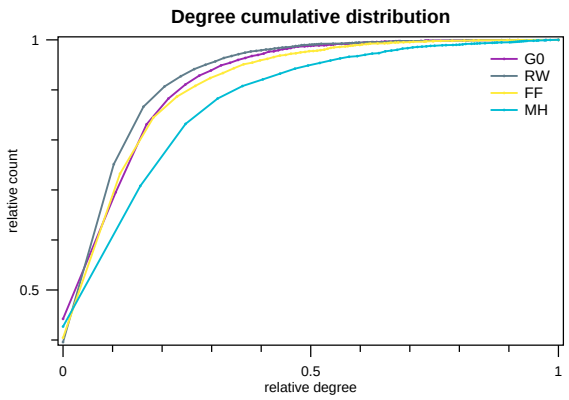
Tabulka 8: Časové okamžiky pro experiment nad datovou sadou AS

Označení	Datum
t_1	1997-12-06
t_2	1998-02-02
t_3	1998-03-18
t_4	1998-06-14
t_5	1998-08-12
t_6	1998-08-22
t_7	1998-11-17
t_8	1998-12-16
t_9	1999-08-06
t_{10}	1999-10-03

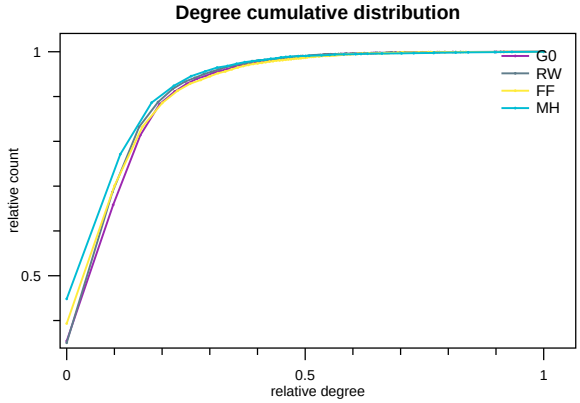
Tabulka 9: Průměrné D-hodnoty nad sítí AS

Method	deg	clus	comp	close	betw	avg
Random Walk	0.401	0.116	0	0.113	0.228	0.172
Forest Fire	0.406	0.219	0	0.217	0.205	0.229
Metropolis	0.508	0.329	0	0.313	0.198	0.269
Random Jump	0.401	0.095	1	0.091	0.254	0.368
Top Degree	0.387	0.131	1	0.127	0.259	0.381
Random Node	0.342	0.164	1	0.174	0.367	0.409
Random Edge	0.402	0.399	1	0.114	0.258	0.435

Na obrázcích 32 a 33 je znázornění kumulativní distribuce stupňů pro metody FF, MHRW a RW. Z grafu je patrné, že všechny tři metody distribuci velmi dobře napodobují.



Obrázek 32: Distribuce stupňů v čase t_2



Obrázek 33: Kumul. dist. stupňů v čase t_8

5.4.7 Srovnání napříč datovými sadami

V této části jsou shrnuty výsledky z předchozích experimentů a srovnání metod napříč datovými sadami. V tabulce 10 jsou výsledky jednotlivých metod, kde každý sloupec je přiřazen jedné datové sadě a obsahuje průměrnou D-hodnotu pro všechny časové okamžiky nad danou sadou. Poslední sloupec obsahuje průměrnou hodnotu z průměrných D-hodnot.

Tabulka 10: Průměrné D hodnoty nad různými daty

GraphName	BA	Random	IMDB	Fb-like	AS	Avg
Forest Fire	**0.1194**	0.1935	0.425	0.086	0.229	**0.211**
Top Degree	0.2309	0.2091	0.371	**0.057**	0.381	0.25
Random Walk	0.1416	0.4481	0.432	0.067	**0.172**	0.252
Random Node	0.3375	**0.0688**	**0.348**	0.289	0.409	0.29
Metropolis	0.2271	0.4498	0.434	0.115	0.269	0.299
Random Jump	0.4049	0.2969	0.405	0.282	0.368	0.351
Random Edge	0.5875	0.1822	0.494	0.316	0.435	0.403

Nejnižší průměrné D hodnoty nad různými daty dosahuje metoda Forest Fire následovaná metodou Top Degree a Random Walk.

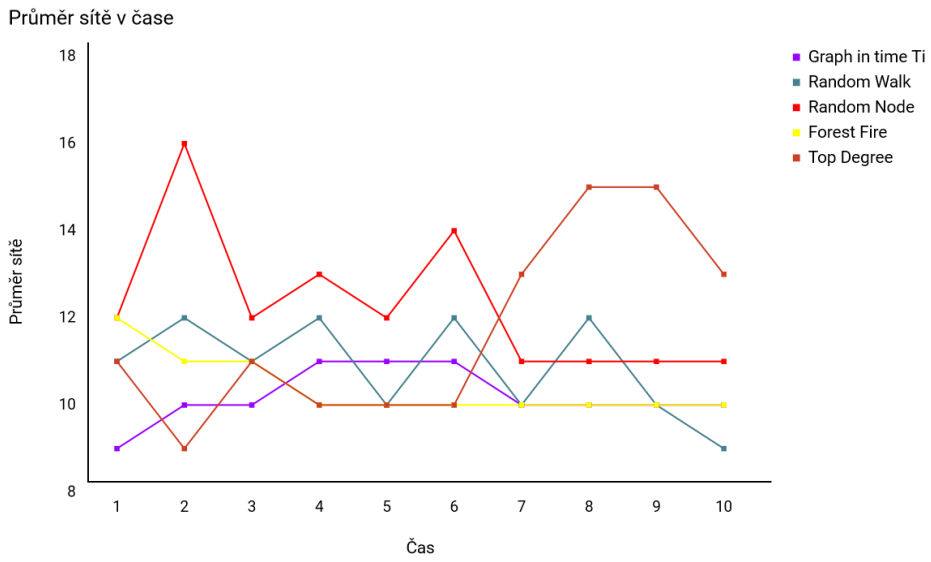
5.4.8 Vlastnosti v čase

Dosavadní experimenty primárně sledovaly jak dobře algoritmy zachovávají vlastnosti, reprezentovány pomocí distribucí, v porovnání k síti v čase t_i. Cílem tohoto experimentu je sledovat vývoj globálních vlastností v čase. Sledované vlastnosti jsou průměr sítě, assortativita a modularita. Pro tyto účely byla použita datová sada AS s 10 náhodnými časovými okamžiky z experimentu 5.4.6 .

Tabulka 11: Průměr sítě v čase

Metoda	Průměrný průměr sítě v čase
Graph in time t_i	10.2
Forest Fire	10.4
Random Walk	10.9
Top Degree	11.7
Random Node	12.3
Random Jump	13.5
Metropolis	13.7
Random Edge	26.9

Metody ForestFire a Random Walk nejlépe zachovává vývoj průměru sítě s ohledem na průměrný průměr sítě v čase (viz. tabulka 11) tak i distribuci průměru v čase znázorněnou na obrázku 34. Graf znázorňuje pouze 3 nejlepší metody z tabulky 11. Fialová křivka sítě G_0 v čase t_i se od času $t_i = 7$ překrývá s křivkou metody FF. Metoda Top Degree i přes 3. nejlepší průměrný průměr sítě v čase, nekopíruje distribuci. Metoda Ranom Edge se jeví jako zcela nejhorší .



Obrázek 34: Vývoj průměru sítě v čase

V tabulce 12 jsou výsledky pozorování průměrné asortativity a modularity v čase. Pro porovnání je nutné sledovat rozdíl hodnot vzhledem k síti v čase t_i. Pořadí nejlepších metod je pro obě vlastnosti stejné, nejlépe si vedla metoda Forest Fire spolus s Random Walk. Nejhorší opět skončila metoda Random Edge.

Tabulka 12: Průměrná assortativita a modularita v čase

Metoda	Assortativita	Modularita
Metropolis	-0.2214	0.6998
Top Degree	-0.211	0.6063
Random Walk	-0.2069	0.6023
Graph in time t_i	-0.1914	0.6409
Forest Fire	-0.1807	0.6653
Random Jump	-0.1767	0.7009
Random Node	-0.175	0.7091
Random Edge	-0.1438	0.9067

5.4.9 Parametry vzorkovacích metod

Nad datovou sadou AS byly testovány algoritmy FF, RW, RJ a jejich parametry. S uniformní pravděpodobnostní byly zvoleny tři časové okamžiky. Nad každým spouštěn algoritmus s různými parametry, výsledek následně vyhodnocen pomocí průměrné D hodnoty.

U metody Forest Fire jsme testovali parametr dopředného zapálení p_f, který jsme testovali v intervalu <0.2,0.8>. Dle pozorování vycházelo, že nad malou požadovanou velikostí vzorku vzhledem k vzorkované sadě, dosahuje algoritmus lepších výsledků s vyšší hodnotou p_f. Naopak v případě velké požadované velikosti vzorku se optimální hodnota pohybovala okolo 0.2. Tento výsledek potvrdil experimenty J. Leskovece [5]. Všechny experimenty byly prováděny s $p_f = 0.25$.

Metoda Random Jump obsahuje parametr c, který označuje pravděpodobnost přeskočení do jiného vrcholu, kde začne náhodná procházka. Metoda generovala nejlepší výsledky v případě, že se hodnota blížila $c = 0.2$. Konečná hodnota parametru byla nastavena na $c = 0.15$.

Metoda Random Walk obsahuje parametr c, který označuje pravděpodobnost návratu do počátečního vrcholu. S vyšším parametrem c jsme pozorovali zvýšenou časovou náročnost algoritmu a zvýšenou hustotu výsledného vzorku. Nad testovanými daty velikost parametru neměla zásadní vliv na výsledné D-hodnoty. Konečná hodnota parametru byla nastavena na $c = 0.15$.

6 Závěr

Cílem práce bylo seznámení s problematikou vzorkování nad rozsáhlými síťovými daty pro účely vzorkování metodou Back-In-Time. Cílem implementační části bylo vytvořit aplikaci, do které budou vstupovat data a různými algoritmy budeme provádět vzorkování. Experimentální část této práce měla za cíl provést, s využitím aplikace, vzorkování nad různými datovými sadami a vyhodnotit metody na základě kvality vzorků.

V rámci práce byly popsány základy teorie grafů a vlastnosti, které u sítí sledujeme. Dále byly popsány jednotlivé algoritmy. Vznikla aplikace, která provede vzorkování dat a přehledně zobrazí vlastnosti vzorků a vyhodnocení jejich kvality. Experimentální část obsahuje sadu experimentů pro získání nejlepší vzorkovací metody za pomocí statických vlastností sítí, D-hodnot a vizuálního porovnání distribucí vlastností. Závěry vyplývající z provedených experimentů:

- Průměrně nejnižší D-hodnotu měly vzorky generované metodami Forest Fire a Random Walk. Velice zajímavé byly výsledky metody Top Degree, která konstantně podávala dobré výsledky a to i nad náhodnou sítí. Nutno zmínit, že oproti metodě FF nebo RW vyžaduje přístup k celé síti, to její využití v praxi komplikuje.

- Nejhorší výsledky jednoznačně generovala metoda Random Edge, která měla nejvyšší D-hodnotu nad všemi analyzovanými datovými soubory (mimo náhodný graf, kde si vedla průměrně).

- Metoda Forest Fire podávala nejlepší výsledky při hodnotě parametru blížící se $p_f = 0.2$, Random Jump s parametrem $c = 0.15$ a metoda Random Walk se ukázala jako velice variabilní a parametr neměl příliš velký dopad na kvalitu vzorku (ale spíše na výkon daného algoritmu). Jako nejlepší byl pro metodu RW zvolen parametr $c = 0.15$.

Aplikace obsahuje velký prostor a potenciál pro zlepšení. Především by bylo vhodné provést optimalizaci práce s daty, protože v krajních případech aplikace využívá extrémní množství prostředků, například při vytváření distribucí. Tento problém by do značné míry mohla řešit změna technologie nad backednem aplikace nebo jeho přesun na server. Dále by bylo vhodné rozšířit možnosti aplikace o dopočítávání vlastností a metod, které při prvním běhu aplikace uživatel vynechal. Případně by bylo určitě zajímavé rozšířit možnosti propojení s prostředím R pro získání většího množství vlastností sítě.

Literatura

[1] BARABÁSI, Albert-Laszlo a Marton POSFAI. *Network science*. Cambridge, United Kingdom: Cambridge University Press, 2016. ISBN 978-110-7076-266.

[2] KOVÁŘ, Petr. *Úvod do Teorie grafů* [online]. 2016 [cit. 2018-04-19]. Dostupné z: <http://homel.vsb.cz/ kov16/files/uvod_do_teorie_grafu.pdf>

[3] BLONDEL V. D., GUILLAUME J.-L., lAMBIOTTE R. a LEFEBVRE E.. *Fast unfolding of communities in large networks*. J. Stat. Mech., 2008.

[4] LESKOVEC J., KLEINBERG J. A FALOUTSOS C.. *Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations*. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2005.

[5] LESKOVEC, Jure a Christos FALOUTSOS. *Sampling from large graphs.* KDD-2006: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining [online]. New York, NY: ACM Press, 2006, s. 631-636 [cit. 2017-01-29]. ISBN 1595933395. Dostupné z: https://cs.stanford.edu/people/jure/pubs/sampling-kdd06.pdf

[6] ERDŐS, Paul a Alfréd RÉNYI. *On random graphs*. I. Publ. Math. Debrecen, 1959, , 290-297.

[7] GILBERT, E.N., *Random Graphs*. Annals of Mathematical Statistics. 30 (1959), no. 4, 1141–1144.

[8] ALBERT, Réka a Albert-László BARABÁSI. *Statistical mechanics of complex networks. Reviews of Modern Physics*. 2002, 74(1), 47-97. ISSN 0034-6861. Dostupné z: https://link.aps.org/doi/10.1103/RevModPhys.74.47

[9] NICOSIA, Vincenzo, John TANG, Cecilia MASCOLO, Mirco MUSOLESI, Giovanni RUSSO a Vito LATORA. *Graph Metrics for Temporal Networks. Temporal Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, 2013-4-15, , 15-40. Understanding Complex Systems. ISBN 978-3-642-36460-0.

[10] *Number of monthly active Facebook users worldwide as of 4th quarter 2017*. [online]. 2018 [cit. 2018-04-19]. Dostupné z: https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/

[11] SLANINA, František a Miroslav KOTRLA. *Sítě „malého světa“: Proč mají odlišné sítě podobnou strukturu?* Vesmír. 2001, 2001(11), 611-614. Dostupné také z: http://casopis.vesmir.cz/files/file/fid/1198/aid/4791

[12] CLAUSET, Aaron. *Network Analysis and Modeling,: Lecture 8*. CSCI 5352 [online]. 2014, 10 [cit. 2018-04-19].

[13] *Dataset: Contacts in a workplace.* Sociopatterns [online]., 2008 [cit. 2018-04-19]. Dostupné z: http://www.sociopatterns.org/datasets/contacts-in-a-workplace/

[14] OPSAHL, Tore *Datasets. Tore Opsahl* [online]. .: Tore Opsahl, 2003 [cit. 2018-04-19]. Dostupné z: https://toreopsahl.com/datasets/#online_social_network

[15] LESKOVEC, Jure. *Autonomous systems AS-733.* SNAP [online], 2012 [cit. 2018-04-19]. Dostupné z: https://snap.stanford.edu/data/as.html

[16] GROSSMAN, Robert. *KDD-2005: proceedings of the eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 21-24, 2005, Chicago, Illinois, USA. New York, NY: ACM Press, c2005. ISBN 15-959-3135-X.

[17] UNGAR, Lyle., Mark. CRAVEN, Dimitrios GUNOPULOS a Tina. ELIASSI-RAD. *KDD-2006: proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*: August 20-23, 2006, Philadelphia, PA, USA. New York, NY: ACM Press, 2006. ISBN 15-959-3339-5.

[18] DAVIDSON, Zeke, Marion VALEIX, Freya VAN KESTEREN, Andrew J. LOVERIDGE, Jane E. HUNT, Felix MURINDAGOMO, David W. MACDONALD a Matt HAYWARD. *Seasonal Diet and Prey Preference of the African Lion in a Waterhole-Driven Semi-Arid Savanna.* PLoS ONE. 2013, 8(2), e55182-. ISSN 1932-6203. Dostupné také z: http://dx.plos.org/10.1371/journal.pone.0055182

[19] MAO, Guoyong and Ning ZHANG, *Analysis of Average Shortest-Path Length of Scale-Free Network*, Journal of Applied Mathematics, vol. 2013, Article ID 865643, 5 pages, 2013.

[20] FLOYD, Robert W., *Algorithm 97: Shortest Path.* Commun. ACM, New York, NY, USA, 1962, 5(6). ISSN 00010782.

[21] HOLME, Petter a Jari SARAMÄKI. *Temporal networks.* New York: Springer, 2013. ISBN 978-3-642-36460-0.

[22] HASAN, Mohammad. *Methods and Applications of Network Sampling.* (2016).

[23] DIESTEL R., *GraphTheory.* GraduateTextsinMathematics, vol.173, Springer-Verlag, Heidelberg (2005).

[24] DARLAY Julien, BRAUNER Nadia, MONCE Julien, *Dense and sparse graph partition* GraduateTextsinMathematics,vol.173,Springer-Verlag,Heidelberg,2005.

[25] OCHODKOVÁ, Eliška. *Metody analýzy dat III* [online]. [cit. 2018-04-19]. Dostupné z: http://www.cs.vsb.cz/ochodkova/

[26] KUDĚLKA, Miloš. *Metody analýzy dat* [online]. [cit. 2017-01-25]. Dostupné z: http://homel.vsb.cz/ kud007

[27] *Statistics - Kolmogorov Smirnov Test.* Tutorialspoint [online]. [cit. 2018-04-19]. Dostupné z: https://www.tutorialspoint.com/statistics/kolmogorov_smirnov_test.htm

[28] AL HASAN, Mohammad, Nesreen AHMED a Jennifer NEVILLE. *Network Sampling: Methods and Applications* [online]. In: . Chicago, 2013 [cit. 2017-02-14]. Dostupné z: https://www.cs.purdue.edu/homes/neville/courses/kdd13-tutorial.html

[29] HASTINGS, W. K. *Monte Carlo sampling methods using Markov chains and their applications.* Biometrika. 1970, 57(1), 97-109. ISSN 1464-3510. Dostupné také z: https://academic.oup.com/biomet/article/57/1/97/284580

[30] WUCHTY S., RAVASZ E., BARABÁSI AL. *The Architecture of Biological Networks.* Deisboeck T.S., Kresh J.Y. (eds) Complex Systems Science in Biomedicine. Topics in Biomedical Engineering International Book Series. Springer, Boston, MA. 2006

[31] *Polinode* [online]. Sydney: Polinode Pty, 2014 [cit. 2018-04-23]. Dostupné z: https://www.polinode.com/

[32] *Graph Theory | Bricklayer 501(c)(3)* [online]. [cit. 2018-04-19]. Dostupné z: <https://bricklayerdotorg.wordpress.com/graph-theory/>

[33] *Simulations in Statistical Physics: Course for MSc physics students* [online]. 2014 [cit. 2018-04-19]. Dostupné z: <http://www.phy.bme.hu/ torok/tanit/SzamSzim/ea11.pdf>

[34] *Protein-protein interaction networks* [online]. 2018 [cit. 2018-04-19]. Dostupné z: <https://www.ebi.ac.uk/training/online/course/network-analysis-protein-interaction-data-introduction/protein-protein-interaction-networks>

[35] *University of Oregon Route Views Project* [online]. 2018 [cit. 2018-04-19]. Dostupné z: <http://www.routeviews.org>

A Příloha na CD/DVD

- **dp.pdf** - tento dokument v elektronické podobě
- **data** - adresář obsahující data použitá pro experimenty
- **exe** - spustitelná aplikace
- **src** - zdrojové kódy aplikace